Efficient Composition of Discrete Time Quantum Walks

by

Xingliang Lou

A thesis

presented to the University of Waterloo

in fulfilment of the

thesis requirement for the degree of

Master of Mathematics

in

Combinatorics and Optimization – Quantum Information

Waterloo, Ontario, Canada, 2017

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

It is well known that certain search problems are efficiently solved by quantum walk algorithms. Of particular interest are those problems whose efficient solutions involve nesting of search algorithms. The nesting of search algorithms generally incurs an extra logarithmic factor in query and time complexity, due to the use of majority voting as an error reduction technique. We study whether composition of search algorithms can be achieved without the said logarithmic factor in complexity. Two methods of composition have been proposed in this thesis. The first is a slight simplification of the quantum walk algorithm due to Magniez, Nayak, Roland, and Santha. The second is a method for composition of Markov chains. Neither approach appear to be generally applicable to all cases of quantum walk composition. Further work is required to determine the circumstances under which these approaches provide the logarithmic factor of increase in efficiency that we desire.

Acknowledgements

I would like to acknowledge the Department of Combinatorics and Optimization and the Institute for Quantum Computing at the University of Waterloo for providing the unique research opportunity that culminated in the writing of this thesis. It would not be possible for me to attain my current level of understanding of quantum computation without the interaction with faculty members and students, of which there are too many to name individually.

I thank Dr. Michele Mosca and Dr. Richard Cleve for agreeing to read my thesis. Furthermore, I am especially grateful towards my supervisor, Dr. Ashwin Nayak, both for his direction and guidance in research, as well as for his understanding and support. The pursuit of excellence in higher education has not been easy, and my supervisor has seen me through the most trying times.

Table of Contents

1 Introduction

1.1 Quantum Walk Search with Imperfect Oracle

Suppose we have a function $f : X \mapsto \{0, 1\}$, where X is finite. We would like to *decide* whether there exists x that is *marked*, i.e., $f(x) = 1$, and *find* such an x if there is.

It is well known that, given a quantum *oracle*, a.k.a. a *black box*, that computes f, the Grover search algorithm efficiently finds such an element with high probability [Gro96]. Its query and time complexities are of order $O(\sqrt{n})$ and $O(\sqrt{n}\log n)$ respectively, assuming that a query to the oracle itself requires only constant time. This is already a significant improvement over the best possible classical algorithm, but the story of quantum speedups does not end here. Sometimes, the structure of the problem allows for even faster searching. Take, for example, the Element Distinctness problem. Invoking Grover's algorithm here amounts to searching over all pairs of elements for a colliding pair, resulting in a query complexity of $O\big(\sqrt{n^2}\big) = O(n)$. However, it has been shown that using a so-called *quantum walk* over the Johnson graph, a query complexity of $O(n^{2/3})$ can be achieved [Amb07]. Advanced search algorithms that offer additional speedup over Grover are typically more complex in structure. For instance, in solving the Triangle Finding problem, there is an algorithm with query complexity $\tilde{O}(n^{13/10})$ that involves using a quantum walk algorithm, where the oracle is a Grover search over yet another quantum walk [MSS07]. The fastest known algorithm for this problem to date in terms of query complexity is due to Le Gall. It achieves a query complexity of $\tilde{O}(n^{5/4})$ with even more layers of nesting of search algorithms [LG14]. Both algorithms are superior to Grover search over all triples of vertices, which results in a query complexity of $O(\sqrt{n^3}) = O(n^{3/2})$.

While the speedup of these algorithms are undoubtedly attractive, they share a common undesirable characteristic. Quantum searching, be it via Grover or quantum walks, typically succeed with high probability, rather than with certainty. In other words, quantum searching has a non-zero probability of *error*. If quantum searching was to be used as an oracle of another search algorithm, the error will accumulate with each call to the oracle. This is a problem, because the analyses of search algorithms such as Grover search and quantum walks typically assume that the oracle is perfect. In quantum information processing, it is well known that error accumulates at most additively. That is, if $U_1, U_2, \ldots, U_n$ and $V_1, V_2, \ldots, V_n$ are unitary operators, and $\|U_j - V_j\| < \gamma$ for all j, then $\|U_1U_2\cdots U_n - V_1V_2\cdots V_n\| < n\gamma$ [NC10]. Hence, suppose we have an imperfect oracle V and a perfect oracle U. If we invoke V multiple times, say k times, in an algorithm instead U, it had better be the case that $\|U - V\| < \gamma/k$, for some small constant γ, so that the entire algorithm succeeds with high probability.

In previous works, the method of obtaining an oracle with such a small error has been via the method of *majority voting*. Here, instead of using the imperfect oracle V as-is, we

invoke V many times over the same input, and use the output that appears more often. It can be shown that if the oracle gives the correct answer with amplitude bounded away from $\frac{1}{\sqrt{2}}$ by a constant, it suffices to take the majority vote over $O(\log(k))$ instances of V to obtain an error of at most ϵ/k. The obvious downside is that this new oracle is built using $O(\log(k))$ instances of the original oracle, and this logarithmic penalty appears in the query and time complexity of the algorithm. In fact, the definition of $\tilde{O}(\cdot)$ with the tilde on the O, which appeared in the query complexities of the above mentioned Triangle Finding algorithms, is $O(\cdot)$ with polylogarithmic factors.

The purpose of this thesis is to examine whether composition of quantum walk algorithms necessarily incur the logarithmic overhead in complexity from majority voting. That is, we are seeking new and possibly more natural ways of composing search algorithms in the hope of removing the logarithmic factor from all quantum walk based search algorithms where composition is involved. Other than the work presented in this thesis, there have been two existing research directions that attempts to accomplish this goal. Many problems that can be solved efficiently using discrete time quantum walks can also be formulated in terms of *span programs* and *learning graphs* [Rei09, Bel12]. There are well-known algorithms for determining whether an oracle computes a *1-input* to a span program or a learning graph. Further, there are known methods of composing span programs and learning graphs such that the query complexity of the resulting algorithm increases only multiplicatively, without any extra logarithmic factors. However, the two said approaches are not without their downsides. With regards to span programs, the quantum algorithm that computes it requires the implementation of a general unitary over a high dimensional Hilbert space. In the worst case, the time complexity of a general unitary is polynomial in the dimension of the Hilbert space, which is exponential in the number of qubits. In the case of learning graphs, the method of composition is limited to the composition of 1-certificates. Ideally, we would like to be able to place a reasonable upper bound on the time complexity as well as the query complexity, and also be able to perform composition more generally. Motivated by these desires, we have chosen to take on this new research direction of composition of discrete time quantum walks, to determine the extent to which we can improve on the existing approaches.

1.2 Background Assumed and Notations Used

Background knowledge in quantum computation is assumed from the reader. Reference texts such as [NC10] and [KLM07] are recommended for readers who lack such background.

In all of our analyses, quantum states, denoted using Dirac's bra-ket notation as in $|\phi\rangle$, are represented as complex vectors in $\mathbb{C}^m$, where m is clear from context. We work only with pure states. The tensor product state $|\phi\rangle \otimes |\psi\rangle$ is sometimes denoted as $|\phi\rangle|\psi\rangle$. All state evolutions are unitary. Measurements, whether partial or complete, are in the computational basis.

Throughout the thesis, the symbol i denotes the imaginary unit $\sqrt{-1}$, unless its meaning is otherwise bounded by its context, e.g. when it appears as a dummy variable in a summation. Following the notational conventions in [MNRS11], the symbol π takes on three meanings. The first is the standard circumference-to-diameter ratio, which is approximately equal to 3.14159. The symbol π generally assumes this meaning when we discuss angles or phases of a complex number. The second is the stationary distribution of a Markov chain, as defined in Definition 3.2.4. In this context, π is a row vector. The third is the initial state of the discrete time quantum walk algorithm $|\pi\rangle$, as defined in Algorithm 3.3.5(1). Only when we surround π with Dirac's bra or ket as in $\langle\pi|$ or $|\pi\rangle$ does it take on this meaning. Similarly, the symbol * takes on two meanings. When it is applied to complex numbers, vectors, and most matrices, * refers to the conjugate transpose of the said object. The one exception to this rule is when it is applied to a reversible, stochastic matrix. In this case, * refers to the time reversal of the said matrix, as defined in Definition 3.2.10.

1.3 Outline of Sections

Since this thesis is meant to be a self-contained treatise on composition of quantum walks, Chapters 2 and 3 are dedicated to background information. In particular, Chapter 2 analyzes the behaviour of product of reflections—an algorithmic component found both in discrete time quantum walks and, as a special case, in Grover search. Chapter 3 outlines relevant properties of Markov chains, and introduces the discrete time quantum walk algorithm due to Magniez, Nayak, Roland, and Santha [MNRS11]. Finally, Chapters 4 and 5 contain detailed discussions on possible approaches of composing quantum walk algorithms without incurring the logarithmic overhead in complexity.

2 Principal Angles and Vectors of Two Subspaces

Many existing analyses of the Grover search algorithm make the observation that in essence, the Grover search iterate is a product of two reflections. The diffusion operator is a reflection about the uniform superposition of all elements, namely, $\sum_{x \in X} |x\rangle$ normalized. Meanwhile, the checking operator reflects about the superposition unmarked elements, which is $\sum_{x \in X, f(x)=0} |x\rangle$ normalized. Throughout the entire algorithm, the state of the system remains within the span of these two states. By a straightforward reasoning in 2D geometry, one obtains that the product of these two reflection operators effects a rotation by an angle of 2θ, where θ is the angle between the two said states after normalization.

The discrete time quantum walk that we will analyze also involves the product of reflections. Though unlike in Grover search, these reflections are not about two states but instead about two *subspaces*. The analysis requires the concept of *principal angles* and *vectors* between two subspaces. This concept was first discovered by Camille Jordan in 1875 [Jor75] and since then it has seen applications in other fields of study such as computer vision and mathematical physics. An overview of the results relevant to discrete time quantum walks will be provided in this chapter. See [Gal08] for a more complete review on the subject.

We will use * to denote the conjugate transpose of a vector or a matrix. Also, we define the dot product $u \cdot v$ between two vectors u and v to be $u^* v$.

2.1 Definition

Let M_1, M_2 be subspaces in $\mathbb{C}^n$, with dimensions k, ℓ respectively. Suppose that $k \geq \ell$. (The analysis for the case where $k < l$ will be omitted as it is very similar.)

Definition 2.1.1 The *principal angles* $\theta_1, \theta_2, \ldots, \theta_\ell$ and *principal vectors* $u_1, u_2, \ldots, u_\ell$, $v_1, v_2, \ldots, v_\ell$ between M_1 and M_2 are angles and vectors such that the following relations hold.

$$\cos\theta_1 = \max_{\substack{u\in M_1, v\in M_2 \\ ||u||=||v||=1}} |u \cdot v| = u_1 \cdot v_1,$$

$$\cos\theta_j = \max_{\substack{u\in M_1, v\in M_2 \\ ||u||=||v||=1 \\ u\perp u_1,\ldots,u_{j-1} \\ v\perp v_1,\ldots,v_{j-1}}} |u \cdot v| = u_j \cdot v_j \qquad \text{for all } 2 \leq j \leq \ell.$$

Further, the *jth principal subspace* is $\text{span}\{u_j, v_j\}$, for all $1 \leq j \leq \ell$.

In this definition, u_j and v_j are vectors that achieves the maxima. Intuitively, the first pair of principal vectors u_1 and v_1 are chosen such that u_1 is in M_1, v_1 is in M_2, and the angle between them is minimized—thus the cosine of the angle is maximized. Then, for all $j \geq 2$, u_j and v_j are again chosen to have minimal angle but with the constraint that u_j is orthogonal to all the previous vectors u_1 through u_{j-1}, and similarly for v_j. We choose the vectors so that $u_j \cdot v_j$ is real and $u_j \cdot v_j \geq 0$. We also choose θ_j to be between 0 and $\pi/2$. That is, $\theta_j = \cos^{-1}(u_j \cdot v_j)$. This way, the principal angles $\theta_1, \ldots, \theta_\ell$ are uniquely determined. The principal vectors are not.

Notice that $M_2 = \text{span}\{v_1, \ldots, v_\ell\}$. This is because $\dim M_2 = \ell$ and $v_1, \ldots, v_\ell$ are orthonormal and therefore linearly independent. However, in the case where $k > \ell$, $\{u_1, \ldots, u_\ell\}$ does not span M_1—there are not enough vectors. In the analysis that follows, we will often want an orthonormal basis for M_1. This motivates the following definition.

Definition 2.1.2 Let $d = k - \ell$. Then let $u_{\ell+1}, u_{\ell+2}, \ldots, u_{\ell+d}$ be unit vectors so that $\{u_1, u_2, \ldots, u_{\ell+d}\}$ form an orthonormal basis for M_1.

Since we now have $M_1 = \text{span}\{u_1, \ldots, u_k\}$, $M_2 = \text{span}\{v_1, \ldots, v_\ell\}$, we can say that $\text{span}\{M_1, M_2\} = \text{span}\{u_1, \ldots, u_k, v_1, \ldots, v_\ell\}$. (Though $\dim \text{span}\{u_1, \ldots, u_k, v_1, \ldots, v_\ell\}$ is not necessarily $k+\ell$.) Later on, we will wish to construct an orthonormal basis for the entire space of $\mathbb{C}^n$. To do that, we need a basis for the orthogonal complement of $\text{span}\{M_1, M_2\}$. This brings us to our last definition.

Definition 2.1.3 Let $e = n - \dim \text{span}\{M_1, M_2\} = \dim \text{span}\{M_1, M_2\}^\perp$. Then let $w_1, w_2, \ldots, w_e$ be unit vectors so that $\{w_1, w_2, \ldots, w_e\}$ form an orthonormal basis for $\text{span}\{M_1, M_2\}^\perp$.

2.2 Basic Properties

Property 2.2.1 $\theta_1 \leq \theta_2 \leq \ldots \leq \theta_\ell$.

Proof. If $\theta_m > \theta_{m+1}$, then u_m, v_m would not be the mth principal vectors since the angle between u_{m+1} and v_{m+1} is smaller. □

Property 2.2.2 $u_i \cdot v_j = \begin{cases} \cos\theta_j & \text{if } i = j, \\ 0 & \text{if } i \neq j, \end{cases}$ for all $1 \leq i \leq k$ and $1 \leq j \leq \ell$.

Proof. If $i = j$, then $u_i \cdot v_j = \cos\theta_j$ by definition. The proof for the case where $i \neq j$ can be found in Theorem 8 of [GH06]. It is a tedious but straightforward inductive proof using the Cauchy-Schwarz inequality. □

Because $u_i \cdot v_j = 0$ whenever $i \neq j$, we now know that $\text{span}\{u_1, v_1\}$, $\text{span}\{u_2, v_2\}$, …, $\text{span}\{u_\ell, v_\ell\}$, $\text{span}\{u_{\ell+1}\}$, …, $\text{span}\{u_k\}$ are mutually orthogonal subspaces. This orthogonality relationship allows us to decompose $\text{span}\{M_1, M_2\}$ as a direct sum of their principal subspaces:

Property 2.2.3
$$\begin{aligned}\text{span}\{M_1, M_2\} = \text{span}\{u_1, v_1\} \oplus \text{span}\{u_2, v_2\} \oplus \cdots \text{span}\{u_\ell, v_\ell\} \\ \oplus \text{span}\{u_{\ell+1}\} \oplus \cdots \oplus \text{span}\{u_k\}.\end{aligned}$$

Our last property concerns how the principal vectors behave under the action of projection matrices onto M_1 and M_2.

Property 2.2.4 Let Π_1, Π_2 be projection matrices onto M_1, M_2. Then $\Pi_1 v_j = \cos\theta_j u_j$ and $\Pi_2 u_j = \cos\theta_j v_j$ for all $1 \leq j \leq \ell$.

Proof. Since Π_1 projects onto M_1, we have $\Pi_1 = \sum_{i=1}^{k} u_i u_i^*$. Then by Property 2.2.2, $\Pi_1 v_j = \sum_{i=1}^{k} u_i u_i^* v_j = \cos\theta_j u_j$. The proof for $\Pi_2 u_j = \cos\theta_j v_j$ is similar. In [GH06], this property is proven simultaneously with Property 2.2.2 in the inductive step. □

2.3 Construction of Orthonormal Basis of $\mathbb{C}^n$ Using Principal Vectors

We know from Section 2.1 that all principal angles are within the interval $[0, \pi/2]$. In this section, we distinguish three kinds of principal subspaces: those with principal angles of 0, strictly between 0 and $\pi/2$, and $\pi/2$. The angles 0 and $\pi/2$ in particular are special for the following reasons.

Property 2.3.1 If $\theta_j = 0$, then $u_j = v_j$; the principal vectors are overlapping.
Hence $\text{span}\{u_j, v_j\} = \text{span}\{u_j\}$.

Property 2.3.2 If $\theta_j = \pi/2$, then $u_j \perp v_j$; the principal vectors are orthogonal.
Hence $\text{span}\{u_j, v_j\} = \text{span}\{u_j\} \oplus \text{span}\{v_j\}$.

We now count the number of principal angles of each kind: let a, b, c respectively denote the number of 0, strictly between 0 and $\pi/2$, and $\pi/2$ principal angles. That is, we have $\theta_1 = \theta_2 = \cdots \theta_a = 0$, $\theta_{a+1}, \ldots, \theta_{a+b} \in (0, \pi/2)$, and $\theta_{a+b+1} = \cdots = \theta_{a+b+c} = \pi/2$. First, we note the following property.

Property 2.3.3 $a = \dim(M_1 \cap M_2)$, and
$M_1 \cap M_2 = \text{span}\{u_1, \ldots, u_a\} = \text{span}\{v_1, \ldots, v_a\}$.

Next, from Property 2.3.1 and Property 2.3.2, we can see that the decomposition of principal subspaces in Property 2.2.3 can be further refined.

Property 2.3.4

$$\begin{aligned}\text{span}\{M_1, M_2\} = &\ \text{span}\{u_1\} \oplus \cdots \oplus \text{span}\{u_a\} \\ &\oplus \text{span}\{u_{a+1}, v_{a+1}\} \oplus \cdots \oplus \text{span}\{u_{a+b}, v_{a+b}\} \\ &\oplus \text{span}\{u_{a+b+1}\} \oplus \text{span}\{v_{a+b+1}\} \oplus \cdots \oplus \text{span}\{u_{a+b+c}\} \oplus \text{span}\{v_{a+b+c}\} \\ &\oplus \text{span}\{u_{a+b+c+1}\} \oplus \cdots \oplus \text{span}\{u_{a+b+c+d}\}.\end{aligned}$$

(Note that $a + b + c = \ell$ and $a + b + c + d = k$. d was defined in Definition 2.1.2.)

The vectors in the above decomposition, namely u_1, …, u_a, u_{a+1}, v_{a+1}, …, u_{a+b}, v_{a+b}, u_{a+b+1}, v_{a+b+1}, …, u_{a+b+c}, v_{a+b+c}, $u_{a+b+c+1}$, …$u_{a+b+c+d}$ form an "almost" orthonormal basis for $\text{span}\{M_1, M_2\}$. The only pairs of vectors that are not orthogonal are u_j and v_j for all j where $1 \le j \le a + b$. In these cases we have $u_j \cdot v_j = \cos\theta_j \neq 0$. All other pairs of vectors are orthogonal either by construction in Definition 2.1.1 or by Property 2.2.2.

To construct an orthonormal basis, we will use the Gram-Schmidt orthogonalization procedure on each of $\{u_{a+1}, v_{a+1}\}$, …, $\{u_{a+b}, v_{a+b}\}$. Recall how this procedure works: we wish to find a vector $u_j^\perp$ such that $\text{span}\{u_j, u_j^\perp\} = \text{span}\{u_j, v_j\}$, where $u_j^\perp$ is orthogonal to u_j. To do so, we subtract from v_j its projection to u_j, and normalize the resulting vector. This leads us to the following definition.

Definition 2.3.5 For all j where $a + 1 \le j \le a + b$, let

$$u_j^\perp = \frac{v_j - u_j u_j^* v_j}{||v_j - u_j u_j^* v_j||} = \frac{v_j - u_j u_j^* v_j}{\sin\theta_j}.$$

The justification for $||v_j - u_j u_j^* v_j|| = \sin\theta_j$ is straightforward: $||v_j - u_j u_j^* v_j||^2 = (v_j - u_j u_j^* v_j)^* (v_j - u_j u_j^* v_j) = v_j^* v_j - v_j^* u_j u_j^* v_j = 1 - \cos^2\theta_j = \sin^2\theta_j$.

An immediate corollary for our definition of $u_j^\perp$ is the following formula for v_j.

Corollary 2.3.6 For all j where $a+1 \le j \le a+b$, $v_j = \cos\theta_j\, u_j + \sin\theta_j\, u_j^\perp$.

Proof. Isolate v_j in Definition 2.3.5, and substitute $u_j^* v_j = u_j \cdot v_j = \cos\theta_j$ by Definition 2.1.1. □

With this definition of $u_j^\perp$, we can now write $\text{span}\{u_j, v_j\} = \text{span}\{u_j, u_j^\perp\} = \text{span}\{u_j\} \oplus \text{span}\{u_j^\perp\}$ for all j between $a+1$ and $a+b$. This leads to our next refinement of Property 2.3.4, now with an orthonormal basis for $\text{span}\{M_1, M_2\}$.

Property 2.3.7

$$\begin{aligned}\text{span}\{M_1, M_2\} = {} & \text{span}\{u_1\} \oplus \cdots \oplus \text{span}\{u_a\} \\ & \oplus \text{span}\{u_{a+1}\} \oplus \text{span}\{u_{a+1}^\perp\} \cdots \oplus \text{span}\{u_{a+b}\} \oplus \text{span}\{u_{a+b}^\perp\} \\ & \oplus \text{span}\{u_{a+b+1}\} \oplus \text{span}\{v_{a+b+1}\} \oplus \cdots \oplus \text{span}\{u_{a+b+c}\} \oplus \text{span}\{v_{a+b+c}\} \\ & \oplus \text{span}\{u_{a+b+c+1}\} \oplus \cdots \oplus \text{span}\{u_{a+b+c+d}\}.\end{aligned}$$

Finally, an orthonormal basis for the entire space of $\mathbb{C}^n$ is simply the union of basis vectors of $\text{span}\{M_1, M_2\}$ and $\text{span}\{M_1, M_2\}^\perp$.

Property 2.3.8

$$\begin{aligned}\mathbb{C}^n = {} & \text{span}\{M_1, M_2\} \oplus \text{span}\{M_1, M_2\}^\perp \\ = {} & \text{span}\{u_1\} \oplus \cdots \oplus \text{span}\{u_a\} \\ & \oplus \text{span}\{u_{a+1}\} \oplus \text{span}\{u_{a+1}^\perp\} \cdots \oplus \text{span}\{u_{a+b}\} \oplus \text{span}\{u_{a+b}^\perp\} \\ & \oplus \text{span}\{u_{a+b+1}\} \oplus \text{span}\{v_{a+b+1}\} \oplus \cdots \oplus \text{span}\{u_{a+b+c}\} \oplus \text{span}\{v_{a+b+c}\} \\ & \oplus \text{span}\{u_{a+b+c+1}\} \oplus \cdots \oplus \text{span}\{u_{a+b+c+d}\} \\ & \oplus \text{span}\{w_1\} \oplus \cdots \oplus \text{span}\{w_e\}.\end{aligned}$$

(e and $w_1, \ldots w_e$ were defined in Definition 2.1.3.)

Counting the number of basis vectors in Property 2.3.8, we see that $n = a + 2b + 2c + d + e$.

2.4 Simultaneous Block Diagonalization of Π_1 and Π_2

As we defined at the end of Section 2.2, let Π_1, Π_2 be projection matrices onto M_1, M_2. A key result in the theory of two subspaces is that Π_1 and Π_2 can be simultaneously block-diagonalized with 1×1 and 2×2 blocks. As we will see now, this diagonalization can be done in the basis of principal vectors. In the text that follows, we will use 1_m and 0_m to respectively denote $m \times m$ identity and zero matrices.

Definition 2.4.1 Let Q be the matrix whose columns are the orthonormal basis vectors for $\mathbb{C}^n$ that we presented in Property 2.3.8. That is,

$$\begin{aligned} Q = [&u_1; \ldots; u_a; \\ &u_{a+1}; u_{a+1}^{\perp}; \ldots; u_{a+b}; u_{a+b}^{\perp}; \\ &u_{a+b+1}; \ldots; u_{a+b+c}; \\ &v_{a+b+1}; \ldots; v_{a+b+c}; \\ &u_{a+b+c+1}; \ldots; u_{a+b+c+d}; \\ &w_1; \ldots; w_e] \end{aligned}$$

Property 2.4.2 (Diagonalization of Π_1)

$$\Pi_1 = Q \begin{bmatrix} 1_a & & & & & & & & \\ & 1 & 0 & & & & & & \\ & 0 & 0 & & & & & & \\ & & & \ddots & & & & & \\ & & & & 1 & 0 & & & \\ & & & & 0 & 0 & & & \\ & & & & & & 1_c & & \\ & & & & & & & 0_c & \\ & & & & & & & & 1_d \\ & & & & & & & & & 0_e \end{bmatrix} Q^*$$

Proof. This diagonalization follows from the fact that every column in Q is either in M_1 or is orthogonal to M_1. So by standard linear algebra, we know that the positions of 1s in the diagonal matrix correspond to the columns in Q that are in M_1—the 1-eigenvectors of Π_1, whereas the position of 0s correspond to the columns in Q orthogonal to M_1—the 0-eigenvectors of Π_1. Looking at the definition of Q in Definition 2.4.1, we see that the first a columns are in M_1, then every odd column in the next $2b$ columns are in M_1, then the next c columns are also in M_1, the c columns after that are *not* in M_1, the next d columns are in M_1, and the last e columns are *not* in M_1. □

Property 2.4.3 (Diagonalization of Π_2)

$$\Pi_2 = Q \begin{bmatrix} 1_a & & & & & & & & & \\ & \cos^2\theta_{a+1} & \cos\theta_{a+1}\sin\theta_{a+1} & & & & & & & \\ & \cos\theta_{a+1}\sin\theta_{a+1} & \sin^2\theta_{a+1} & & & & & & & \\ & & & \ddots & & & & & & \\ & & & \cos^2\theta_{a+b} & \cos\theta_{a+b}\sin\theta_{a+b} & & & & & \\ & & & \cos\theta_{a+b}\sin\theta_{a+b} & \sin^2\theta_{a+b} & & & & & \\ & & & & & 0_c & & & & \\ & & & & & & 1_c & & & \\ & & & & & & & 0_d & & \\ & & & & & & & & 0_e & \end{bmatrix} Q^*$$

Proof. The justification for the first a and the last $2c+d+e$ diagonal entries are the same as that of Property 2.4.2, except we now look at which columns of Q are within M_2: the first a columns of Q are in M_2 due to Property 2.3.1; skipping the next $2b$ columns for now, the next c columns are orthogonal to M_2, the c columns after that are in M_2, and finally the last $d+e$ columns are orthogonal to M_2. The middle $2b$ entries are trickier because $u_{a+1}, u_{a+1}^\perp, \ldots, u_{a+b}, u_{a+b}^\perp$ are neither within nor orthogonal to M_2. But we know instead that $v_j \in M_2$, and Corollary 2.3.6 tells us how to express v_j in the basis of u_j and $u_j^\perp$: $v_j = \cos\theta_j\, u_j + \sin\theta_j\, u_j^\perp$. Therefore, in the basis of u_j and $u_j^\perp$, Π_2 acts like $\begin{bmatrix}\cos\theta_j \\ \sin\theta_j\end{bmatrix}[\cos\theta_j\ \sin\theta_j] = \begin{bmatrix}\cos^2\theta_j & \cos\theta_j\sin\theta_j \\ \cos\theta_j\sin\theta_j & \sin^2\theta_j\end{bmatrix}$. This is the form of the 2×2 block that appears b times in the above matrix—once for each principal subspace of angle strictly between 0 and $\pi/2$. □

For conciseness, let Λ_1 and Λ_2 denote the block diagonal matrices in Property 2.4.2 and Property 2.4.3 so that $\Pi_1 = Q\Lambda_1 Q^*$ and $\Pi_2 = Q\Lambda_2 Q^*$. We restate the above diagonalizations of Π_1 and Π_2 as follows.

Property 2.4.4 (Block diagonalizations of Π_1 and Π_2, stated more concisely)

$$\Pi_1 = Q\Lambda_1 Q^* = Q\ \mathrm{diag}\left(1_a, \begin{bmatrix}1 & 0 \\ 0 & 0\end{bmatrix}_{j=a+1,\ldots,a+b}, 1_c, 0_c, 1_d, 0_e\right) Q^*$$

$$\Pi_2 = Q\Lambda_2 Q^* = Q\ \mathrm{diag}\left(1_a, \begin{bmatrix}\cos^2\theta_j & \cos\theta_j\sin\theta_j \\ \cos\theta_j\sin\theta_j & \sin^2\theta_j\end{bmatrix}_{j=a+1,\ldots,a+b}, 0_c, 1_c, 0_d, 0_e\right) Q^*$$

2.5 Sum and Product of Projections, Product of Reflections

Following Property 2.4.4, the block diagonalizations of $\Pi_1+\Pi_2$, $\Pi_1\Pi_2$, and $(2\Pi_1-1_n)(2\Pi_2-1_n)$ are simple corollaries.

Corollary 2.5.1 (Sum of projection matrices)

$$\Pi_1 + \Pi_2 = Q \operatorname{diag}\left(2_a, \begin{bmatrix} 1+\cos^2\theta_j & \cos\theta_j \sin\theta_j \\ \cos\theta_j \sin\theta_j & \sin^2\theta_j \end{bmatrix}_{j=a+1,\ldots,a+b}, 1_c, 1_c, 1_d, 0_e\right) Q^*$$

Corollary 2.5.2 (Product of projection matrices)

$$\Pi_1\Pi_2 = Q \operatorname{diag}\left(1_a, \begin{bmatrix} \cos^2\theta_j & \cos\theta_j \sin\theta_j \\ 0 & 0 \end{bmatrix}_{j=a+1,\ldots,a+b}, 0_c, 0_c, 0_d, 0_e\right) Q^*$$

Corollary 2.5.3 (Product of reflection matrices)

$$(2\Pi_1 - 1_n)(2\Pi_2 - 1_n) = Q \operatorname{diag}\left(1_a, \begin{bmatrix} \cos 2\theta_j & \sin 2\theta_j \\ -\sin 2\theta_j & \cos 2\theta_j \end{bmatrix}_{j=a+1,\ldots,a+b}, -1_c, -1_c, -1_d, 1_e\right) Q^*$$

Proof.

$$\begin{aligned} \Pi_1 + \Pi_2 &= Q\Lambda_1 Q^* + Q\Lambda_2 Q^* &&= Q(\Lambda_1 + \Lambda_2)Q^* \\ \Pi_1\Pi_2 &= Q\Lambda_1 Q^* Q\Lambda_2 Q^* &&= Q(\Lambda_1\Lambda_2)Q^* \\ (2\Pi_1 - 1_n)(2\Pi_2 - 1_n) &= (2Q\Lambda_1 Q^* - QQ^*)(2Q\Lambda_2 Q^* - QQ^*) &&= Q(2\Lambda_1 - 1_n)(2\Lambda_2 - 1_n)Q^* \end{aligned}$$

The reader may verify that the diagonal matrices stated in the three corollaries are indeed equal to $\Lambda_1 + \Lambda_2$, $\Lambda_1\Lambda_2$, and $(2\Lambda_1 - 1_n)(2\Lambda_2 - 1_n)$. □

Corollary 2.5.3 in particular is useful in understanding the behaviour of quantum walks, since the product of reflection operators is a commonly found component in such algorithms. We observe that this operator effects a rotation by $2\theta_j$ in each of the principal subspaces. Most notably, within the principal subspaces with 0 principal angle—that is, within $M_1 \cap M_2$—it acts as the identity operator 1_a. Incidentally, the eigenvalues of the 2D rotation matrix $\left[\begin{smallmatrix} \cos 2\theta_j & \sin 2\theta_j \\ -\sin 2\theta_j & \cos 2\theta_j \end{smallmatrix}\right]$ are $\exp(2\theta_j i)$ and $\exp(-2\theta_j i)$. Their corresponding eigenvectors are $\left[\begin{smallmatrix} 1 \\ i \end{smallmatrix}\right]$ and $\left[\begin{smallmatrix} 1 \\ -i \end{smallmatrix}\right]$. As for Corollary 2.5.2, we will see next that it can be used as a practical means for computing the principal angles and vectors of two subspaces. We do not make any further use of Corollary 2.5.1 in this thesis, but we stated it here anyway because it is as immediate a corollary as the other two.

2.6 Singular Value Decomposition of Product of Projection Matrices

Our last topic of discussion on the subject of principal angles and vectors is the method by which they are computed. While Definition 2.1.1 suggests a maximization problem to which principal vectors are solutions, in practice this definition is not used directly. Instead, an approach commonly used within the quantum literature is the singular value decomposition of $\Pi_1\Pi_2$. We will see below that the singular values of $\Pi_1\Pi_2$ are $\cos\theta_1$, ..., $\cos\theta_\ell$, 0, ..., 0. Further, the left and right singular vectors of the nonzero singular values are the corresponding principal vectors.

Let

$$\Lambda = \operatorname{diag}\left(1_a, \begin{bmatrix} \cos^2\theta_j & \cos\theta_j \sin\theta_j \\ 0 & 0 \end{bmatrix}_{j=a+1,\ldots,a+b}, 0_c, 0_c, 0_d, 0_e\right)$$

so that $\Pi_1\Pi_2 = Q\Lambda Q^*$, as we stated in Corollary 2.5.2. To find the singular value decomposition of $\Pi_1\Pi_2$, we make two observations. First, since Q is unitary, the singular values of $\Pi_1\Pi_2$ and Λ are the same, whereas the singular vectors are related by a change in basis via matrix Q. Second, since Λ is block diagonal, the singular values and vectors of the entire matrix are simply those of each diagonal block. Hence, we examine Λ block by block.

First, all singular values of 1_a are 1. The singular vectors corresponding to this block are the first a columns of Q. These are the principal vectors whose principal angles are 0—in other words, the vectors in $M_1 \cap M_2$. Next, the singular values of 0_c, 0_c, 0_d, and 0_e are all zero. The singular vectors corresponding to these are one of several things: principal vectors whose principal angles are $\pi/2$, the basis vectors in M_1 that are not principal vectors as defined in Definition 2.1.2, and vectors in $\operatorname{span}\{M_1, M_2\}^\perp$ as defined Definition 2.1.3. Lastly, observe that the 2×2 blocks in Λ have the singular value decomposition

$$\begin{bmatrix} \cos^2\theta_j & \cos\theta_j \sin\theta_j \\ 0 & 0 \end{bmatrix} = \cos\theta_j \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} \cos\theta_j & \sin\theta_j \end{bmatrix} + 0 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} \sin\theta_j & -\cos\theta_j \end{bmatrix}.$$

So every such 2×2 block gives rise to two singular values. The first is $\cos\theta_j$. Its left and right singular vectors are $1 \cdot u_j + 0 \cdot u_j^\perp = u_j$ and $\cos\theta_j u_j + \sin\theta_j u_j^\perp = v_j$, i.e., the principal vectors. The other singular value is 0, and its singular vectors are orthogonal complements to u_j and v_j in $\operatorname{span}\{u_j, v_j\}$.

3 Markov Chains and Discrete Time Quantum Walks

The purpose of this chapter is to provide the foundation upon which further discussions on discrete time quantum walks can be built. We begin with a brief discussion on relevant properties regarding Markov chains, because a walk on such a chain is the namesake task that quantum walks perform faster than their classical counterpart. Then, a description of the discrete time quantum walk due to Magniez, Nayak, Roland, and Santha [MNRS11], henceforth referred to as the *MNRS quantum walk*, is given. Finally, as a segue to to our subsequent analyses, we explain more precisely what is meant by a "composed" quantum walk, and discuss the performance of its naive implementation.

3.1 Markov Chains

A *Markov chain* is a step-by-step random process. The state of this process is always some element in a set X called the *state space*. With every step, the state *transitions* from one element of X to another. The probability of this transition depends only on the current and the next state. If $X = \{1, 2, \ldots, n\}$, one can characterize a Markov chain with an $n \times n$ transition matrix P. The entry on the ith row and jth column of P, denoted as p_{ij}, is the transition probability from state i to state j. As a consequence of P being a matrix containing probabilities, every element of P is nonnegative, and every row of P sums to 1. We call a matrix such as P that satisfies these two properties a *stochastic* matrix.

In the context of quantum algorithms, we often associate a Markov chain with a directed graph. If we consider a graph whose set of vertices is X, and contains an edge from vertex i to vertex j iff $p_{ij} > 0$, then an instance of a Markov process is a random path—in other words, a random walk—on this graph, where p_{ij} is the probability of the edge i — j being selected from vertex i.

One can contemplate the problem of finding a *marked state*, or a *marked vertex* in a Markov chain. That is, suppose there is a function $f : X \mapsto \{0, 1\}$ and we would like to find a state $x \in X$ such that $f(x) = 1$. Classically, one strategy would be to pick some element in X as the initial state, and then carry out the Markov process step by step until we land on a marked state. The average number of steps this process takes is the well-studied *expected hitting time*. What makes this problem relevant to us is that there also exist efficient *quantum* algorithms that find a marked element efficiently. Furthermore, other problems such as Element Distinctness and Triangle Finding reduce to instances of finding a marked state in a Markov chain.

The variant of quantum walk algorithm that we study is the MNRS quantum walk [MNRS11]. We first explain the properties that a Markov chain must satisfy for it to be used by this algorithm, and introduce some notation that we use consistently in this thesis.

3.2 Irreducibility, Aperiodicity, Reversibility, δ, ϵ

The MNRS quantum walk does not work on all Markov chains. Its analysis assumes that the Markov chain is irreducible, aperiodic, and reversible. The complexity of the algorithm also depends on quantities such as the spectral gap δ, and the ratio of marked elements ϵ. The definitions of these terms as well as properties relevant to them are stated below. We do not prove all the properties, especially those regarding Markov chains in general. Proofs are provided only for those properties that are motivated by the specific application of quantum walks. In every case where a proof is not provided here, a reference to the proof is given along with the statement of the property.

Definition 3.2.1 A Markov chain is *irreducible* if every state in X can be reached from any state in X; equivalently, an irreducible Markov chain is one where the associated directed graph is strongly connected.

Property 3.2.2 The largest eigenvalue (largest in magnitude) of a stochastic matrix is 1. (See [Spi] or Section 8.7 of [HJ12] for proof.)

Property 3.2.3 If P is the stochastic matrix that characterizes an irreducible Markov chain, then its 1-eigenvector and 1-left-eigenvector are unique up to scalar multiplication. Further, all elements of its 1-eigenvector and 1-left-eigenvector can be chosen to be strictly positive. (See Theorem 1.5 of [Sen06] for proof. This is a special case of the well-known *Perron-Frobenius theorem*.)

Definition 3.2.4 The *stationary distribution* of an irreducible Markov chain, denoted as π, is the unique 1-left-eigenvector of P, normalized so that its elements sum to 1. Let π_1, π_2, ..., π_n denote the elements of π.

Definition 3.2.5 Let $x \in X$. The *period* of a state x is $\gcd\{k \mid P^k_{xx} > 0, k > 0\}$; this is the greatest common divisor of lengths of all paths that goes from x to x. The period is undefined if no such path exists.

Property 3.2.6 The periods of all states in an irreducible Markov chain are the same. (See Lemma 1.2 of [Sen06] for proof.)

Definition 3.2.7 As a consequence of Property 3.2.6, we define the *period of an irreducible Markov chain* to be the period of any one of its states.

Property 3.2.8 If matrix P characterizes an irreducible Markov chain with period d, then the eigenvalues of P with modulus 1 are precisely the dth roots of unity: 1, $\exp(2\pi i/d)$, $\exp(2 \cdot 2\pi i/d)$, ... , $\exp((d-1) \cdot 2\pi i/d)$. (See Theorem 1.7 of [Sen06] for proof.)

Definition 3.2.9 An irreducible Markov chain is *aperiodic* if its period is 1.

Definition 3.2.10 Let P be the matrix that characterizes an irreducible Markov chain. The *time reversal* of P, denoted as P^*, is a matrix whose elements are given by

$$p^*_{yx} = \frac{\pi_x}{\pi_y} p_{xy} \qquad \text{for all } x, y \in X.$$

Definition 3.2.11 A Markov chain characterized by matrix P is *reversible* if it is irreducible and $\pi_x p_{xy} = \pi_y p_{yx}$ for all $x, y \in X$; in other words, if $P = P^*$.

Property 3.2.12 All eigenvalues of an irreducible, reversible Markov chain matrix are real.

Proof. Let P be the stochastic matrix that characterizes an irreducible and reversible Markov chain. Let $U = \text{diag}(\sqrt{\pi_1}, \sqrt{\pi_2}, \ldots, \sqrt{\pi_n})$. One can show, using Definition 3.2.11, that UPU^{-1} is real symmetric. The desired result follows from the fact that all eigenvalues of a real symmetric matrix are real. □

Corollary 3.2.13 The period of an irreducible, reversible Markov chain is either 1 or 2.

Proof. This is immediate from Property 3.2.12 and Property 3.2.8. Only the first and second roots of unity are all real numbers. □

Property 3.2.14 An irreducible, reversible Markov chain is aperiodic iff it is not bipartite. (This property provides a simple way of checking whether a reversible Markov chain is aperiodic.)

Proof. An important property of an irreducible Markov chain with period d is that its state space X can be partitioned into d disjoint subsets $X_0, X_1, \ldots, X_{d-1}$ such that for every $j \in \{0, 1, \ldots, d-1\}$, X_j has at least one element, and all edges from X_j go to $X_{(j+1) \bmod d}$. (See Section 1.3 of [Sen06] for proof.) It can then be shown that the associated graph of an irreducible Markov chain is bipartite if and only if its period is divisible by 2. From here, Corollary 3.2.13 immediately leads to desired result. □

Definition 3.2.15 Let P be the matrix that characterizes an irreducible Markov chain. The *spectral gap* of the Markov chain, denoted as δ, is the difference in magnitude between 1 and the second largest eigenvalue of P in magnitude.

Definition 3.2.16 Suppose there is a function $f : X \mapsto \{0, 1\}$. The goal of the MNRS quantum walk algorithm is to find an x such that $f(x) = 1$. The *ratio of marked elements* of a Markov chain, denoted ϵ, is $\sum_{x \in X, f(x)=1} \pi_x$.

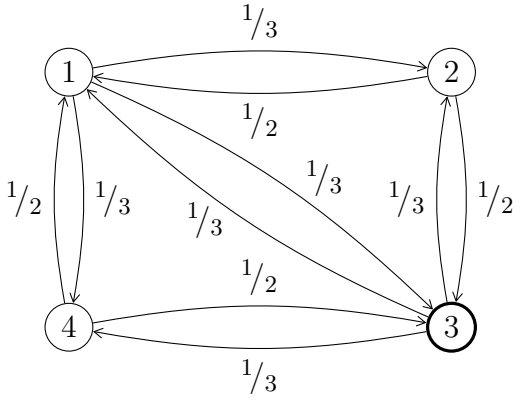
We have introduced a lot of concepts in this section. To help consolidate everything that we know so far, we now illustrate the preceding definitions and properties through an example.

Example 3.2.17 Consider the Markov chain on state space X characterized by matrix P, and the function $f : X \mapsto \{0, 1\}$, defined below.

$$X = \{1, 2, 3, 4\}$$

$$P = \begin{bmatrix} 0 & 1/3 & 1/3 & 1/3 \\ 1/2 & 0 & 1/2 & 0 \\ 1/3 & 1/3 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 0 \end{bmatrix}$$

$$f(1) = 0, f(2) = 0, f(3) = 1, f(4) = 0$$



The eigenvalues and left-eigenvectors of P are as follows.

$$\begin{array}{ll} \lambda_1 = 1 & \frac{1}{10}\begin{bmatrix} 3 & 2 & 3 & 2 \end{bmatrix} \quad (= \pi) \\ \lambda_2 = -2/3 & \begin{bmatrix} -1 & 1 & -1 & 1 \end{bmatrix} \\ \lambda_3 = -1/3 & \begin{bmatrix} 1 & 0 & -1 & 0 \end{bmatrix} \\ \lambda_4 = 0 & \begin{bmatrix} 0 & 1 & 0 & -1 \end{bmatrix} \end{array}$$

First, since P is a stochastic matrix, Property 3.2.2 tells us that 1 is one of its eigenvalues, and no other eigenvalue has magnitude greater than 1. We see that this is indeed the case for P. Then, observe that the associated graph of this Markov chain is strongly connected. This means that the Markov chain is irreducible. As a result, the Perron-Frobenius theorem tells us that the 1-eigenvector is unique, in the sense that the algebraic and geometric multiplicity of this eigenvalue are both 1. Further, there exists a 1-eigenvector with strictly positive elements. These are all true in the case of P. While any scalar multiple of our 1-eigenvector is still a 1-eigenvector, we define the stationary distribution π to be the one we wrote because its elements sum to 1. Next, this Markov chain is reversible. This fact can be checked by verifying directly that $\pi_x p_{xy} = \pi_y p_{yx}$ is true for all $x, y \in X$. Consequently, the eigenvalues of P are real. Lastly, we see that the Markov chain is also aperiodic. There are two ways of verifying this fact. The first way is by observing that the associated graph is not bipartite due to the presence of odd-length cycles such as $1 — 2 — 3 — 1$. Hence one can prove aperiodicity by invoking Property 3.2.14. Another way is by noticing that only the first root of unity, 1, is an eigenvalue of P. A periodic Markov chain would also have other roots of unity as eigenvalues due to Property 3.2.8. Since we already know that the Markov chain reversible, we need to check only that the second root of unity, -1, is not an eigenvalue due to Corollary 3.2.13. Because this Markov chain is irreducible, aperiodic, and reversible, we can use the MNRS quantum walk algorithm to find a marked state.

The performance of the algorithm depends on the spectral gap and the ratio of marked elements. It also depends on other quantities such as the *set-up*, *update*, and *checking*

costs. We will look at these last three quantities in the next section. For the time being, we know that the eigenvalue of P that is second largest in magnitude is $-2/3$. So the spectral gap is $\delta = 1 - |-2/3| = 1/3$. We can also compute the ratio of marked elements: $\epsilon = \sum_{x \in X, f(x)=1} \pi_x = \pi_3 = 3/10$.

3.3 The MNRS Quantum Walk

Below is the problem solved by the MNRS quantum walk.

Problem 3.3.1 (Finding a marked state in a Markov chain)
Suppose there is an irreducible, aperiodic, and reversible Markov chain on some state space X characterized by the transition matrix P. Suppose also that there is a function $f : X \mapsto \{0, 1\}$. If there exists an $x \in X$ such that $f(x) = 1$, find such an x. If not, return "not found."

There are in fact two versions of the MNRS algorithm: a simple and a complex one. The complex version offers a slight improvement in asymptotic complexity. Both of these algorithms assume the existence of the so-called *set-up*, *update*, and *checking* operators, which are defined below.

Definition 3.3.2 (Set-up, update, and checking)
Available to the MNRS algorithm are

(1) The *set-up* operator, U_S, which maps $|0\rangle$ to $\sum_{x\in X} \sqrt{\pi_x}|x\rangle$.

(2) The two *update operators* U_1 and U_2 and their inverses, with actions

$$U_1 : |x\rangle|0\rangle \mapsto \sum_{y\in X} \sqrt{p_{xy}}|x\rangle|y\rangle \qquad \text{and} \qquad U_2 : |0\rangle|y\rangle \mapsto \sum_{x\in X} \sqrt{p^*_{yx}}|x\rangle|y\rangle.$$

(3) The *checking* operator, i.e., the *oracle*, U_C, which maps $|x\rangle$ to $(-1)^{f(x)}|x\rangle$.

Following the notational conventions in [MNRS11], we let S, U, and C denote the respective *costs* of these operators. As in the said paper, we use the word "cost" as a blanket term to mean either the time complexity or the query complexity with respect to some oracle (not necessarily U_C), depending on context. In our analysis, we assume for simplicity that the operators in Definition 3.3.2 act on registers of dimension $|X|$—which is to say that $|x\rangle, |y\rangle \in \mathbb{C}^{|X|}$, and that $|0\rangle$ is some arbitrary initial state in this space. In reality, it is common for the states $|x\rangle$ and $|y\rangle$ to be entangled with some data structure that depends on x and y. If we want to be strictly correct, all occurrences of $|x\rangle$ and $|y\rangle$ in this section really ought to be replaced by $|x\rangle|\text{data}(x)\rangle$ and $|y\rangle|\text{data}(y)\rangle$. The set-up, update, and checking costs should also include the cost of accessing and modifying the data structure. We do not consider the data structure in this thesis, because it distracts us from the main points of discussion. Since there is an isomorphism between the states with data structure, i.e., $|x\rangle|\text{data}(x)\rangle$, and the states without, i.e., $|x\rangle$, it is not difficult to adapt our later proofs to properly account for the presence of data structures.

Letting δ and ϵ denote the spectral gap and the ratio of marked elements of the Markov chain, the performance of the MNRS walk is then as follows.

Theorem 3.3.3 (Simple MNRS, Section 3 of [MNRS11])
There is an algorithm that solves Problem 3.3.1 with cost $O(\mathsf{S} + \frac{1}{\sqrt{\epsilon}}(\frac{1}{\sqrt{\delta}}\mathsf{U}\log\frac{1}{\sqrt{\epsilon}} + \mathsf{C}))$.

Theorem 3.3.4 (Complex MNRS, Section 4 of [MNRS11])
There is an algorithm that solves Problem 3.3.1 with cost $O(\mathsf{S} + \frac{1}{\sqrt{\epsilon}}(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C}))$.

We now detail the steps of the MNRS algorithm. On a high level, both versions follow the same steps. The algorithm uses two data (i.e., non-ancillary) registers and many ancilla registers. Each data register resides in a space of dimension $|X|$.

Algorithm 3.3.5 (The MNRS quantum walk)
(1) Create the state $|\pi\rangle := \sum_{x\in X} \sqrt{\pi_x}|x\rangle \sum_{y\in X} \sqrt{p_{xy}}|y\rangle$ on the two data registers.
Repeat Steps 2 and 3 $O(1/\sqrt{\epsilon})$ times
(2) Flip the phase of marked states on the first register. That is, perform the operation

$$|x\rangle|y\rangle \mapsto \begin{cases} -|x\rangle|y\rangle & \text{if } f(x) = 1, \\ |x\rangle|y\rangle & \text{if } f(x) = 0. \end{cases}$$

(3) Reflect the two registers *approximately* about the $|\pi\rangle$ state.
(4) Measure the first register. Return the measurement outcome if it is a marked state; return "not found" otherwise.

The implementations of Steps 1, 2, and 4 are straightforward using the operators given by Definition 3.3.2. We leave it to the reader to verify that the cost of Step 1 is $\mathsf{S} + \mathsf{U}$, and the costs of Steps 2 and 4 are both C. Step 3 is the most complicated. This is the step whose implementation requires ancilla registers. We remark that, if Step 3 can be implemented *exactly*, the MNRS quantum walk would be identical to Grover's amplitude amplification. However, Step 3 is efficiently implemented using the phase estimation algorithm—an algorithm which is not exact. In the MNRS walk, the precision of Step 3 is chosen so that the behaviour of the entire algorithm differs negligibly from Grover's amplitude amplification.

We discuss below the implementation of Step 3. To avoid repeating much of the analysis in [MNRS11], we do not provide proof for the properties that we claim, though we do mention here that it is the proof of Property 3.3.8 that requires the Markov chain to be reversible and aperiodic.

Definition 3.3.6 A key component of Step 3 of Algorithm 3.3.5 is the *quantum walk operator*, which we denote as U_W. U_W acts on two registers of dimensions $|X|$. It is the product of two reflection operators $(2\Pi_B - I)(2\Pi_A - I)$, where
(1) Π_A is the projector onto the subspace $A := \text{span}\{\sum_{y\in X} \sqrt{p_{xy}}|x\rangle|y\rangle \mid x \in X\}$,
(2) Π_B is the projector onto the subspace $B := \text{span}\{\sum_{x\in X} \sqrt{p^*_{yx}}|x\rangle|y\rangle \mid y \in X\}$,
(3) I is the identity operator.

Property 3.3.7 The cost of U_W is in $\mathrm{O}(\mathsf{U})$.

Property 3.3.8 The largest singular value of $\Pi_B\Pi_A$ is 1. Its unique left and right singular vectors (up to scalar multiplication) are both $|\pi\rangle$. All other singular values of $\Pi_B\Pi_A$ are no greater than $1 - \delta$.

Property 3.3.9 Consequently, by Corollary 2.5.3, $|\pi\rangle$ is a 1-eigenvector of U_W. Eigenvalues of all other eigenvectors of U_W in $\text{span}\{A, B\}$ are of the form $\exp(2\theta i)$, where $\cos^{-1}(1-\delta) \leq |\theta| \leq \pi/2$. Note that $\cos^{-1}(1-\delta) \in O(1/\sqrt{\delta})$.

Theorem 3.3.10 (Phase estimation, Section 5.2 of [NC10] or Section 7.2 of [KLM07]) There exists an algorithm—namely, phase estimation with precision $O(1/\sqrt{\delta})$—with the following properties.

(1) It uses two input registers of dimensions $|X|$.

(2) It uses an output register of dimension $O(1/\sqrt{\delta})$.

(3) It uses $O(1/\sqrt{\delta})$ instances of controlled-U_W, plus a few other auxiliary operations such as Hadamard transformations, controlled phase rotations, and (optionally) swap operations. Consequently, the cost of this algorithm is in $O(\frac{1}{\sqrt{\delta}}\mathsf{U})$.

Further, if we let Φ denote this algorithm, then

(4) $\langle\pi|\langle 0|\Phi|\pi\rangle|0\rangle = 1$.

(If the input state is $|\pi\rangle$, the output register remains $|0\rangle$. Since $|\pi\rangle$ is a 1-eigenvector, phase estimation computes the eigenvalue $1 = \exp(0i)$ *exactly*.)

(5) If $|\phi\rangle \in \text{span}\{A, B\}$ and $\langle\pi|\phi\rangle = 0$, then
$|\langle\phi|\langle 0|\Phi|\phi\rangle|0\rangle| \leq \beta$, where β is some small constant.

(Any eigenvector of U_W that is orthogonal to $|\pi\rangle$ and is still in $\text{span}\{A, B\}$ will *not* be a 1-eigenvector. Its eigenvalue will be in the form $\exp(2\theta i)$, where the phase 2θ is bounded *away* from 0 due to Property 3.3.9. Since phase estimation is not exact, its output may still have a small projection onto $|0\rangle$. The precision of $O(1/\sqrt{\delta})$ ensures that the norm of this projection is bounded above by a small constant.)

We now see that the phase estimation algorithm with precision $O(1/\sqrt{\delta})$ can be used to approximate Step 3 of Algorithm 3.3.5 with constant-bounded error.

Algorithm 3.3.11 (Step 3 of Algorithm 3.3.5)
Given an input state $|\phi\rangle$, we wish to reflect it approximately about $|\pi\rangle$. To do so, we initialize k fresh ancilla registers, each of dimension $O(1/\sqrt{\delta})$, to $|0\rangle$.

(1) Run phase estimation as described in Theorem 3.3.10 k times, using the same input state $|\phi\rangle$ but a different ancilla register as the output register every time. The parameter k is defined in the comments below.

(2) Flip the phase if any of the output registers is non-$|0\rangle$,

(3) Perform the inverse of Step 1.

The reader can check that Algorithm 3.3.11 correctly maps the input state $|\pi\rangle$ to itself and perfectly cleans up all ancilla registers when the input is $|\pi\rangle$. If the input state $|\phi\rangle$ is orthogonal to $|\pi\rangle$ and $|\phi\rangle \in \text{span}\{A, B\}$, then in the case when $k = 1$, phase estimation computes a nonzero phase with probability at least $1 - \beta^2$, where β is as defined in Theorem 3.3.10(5). This implies that Algorithm 3.3.11 has at most constant-bounded error. The error can be made smaller by choosing a bigger k, but the cost of this algorithm will then be $O(k \cdot \frac{1}{\sqrt{\delta}}\mathsf{U})$. It is a well known property that errors in a

quantum circuit accumulate at most additively. (See Box 4.1 of Section 4.5.3 of [NC10].) Since Step 3 of Algorithm 3.3.5 is repeated $O(1/\sqrt{\epsilon})$ times, we need the *cumulative* error from $O(1/\sqrt{\epsilon})$ repetitions of Algorithm 3.3.11 to be bounded by a small constant. The difference between the simple and complex versions of the MNRS walk is the strategy in how this is done. In the simple version, we choose k to be in $O(\log \frac{1}{\sqrt{\epsilon}})$. It can be shown that this many repetitions of phase estimation suffices in suppressing the error of each iteration of Algorithm 3.3.11 to the order of $\beta\sqrt{\epsilon}$. The cost of Algorithm 3.3.11 is then $O(\log(\frac{1}{\sqrt{\epsilon}}) \cdot \frac{1}{\sqrt{\delta}}\mathsf{U})$, giving rise to the $\frac{1}{\sqrt{\delta}}\mathsf{U} \log \frac{1}{\sqrt{\epsilon}}$ term in Theorem 3.3.3. In the complex version, k varies throughout the algorithm. The pattern in which the k varies is recursively defined as to mimic that of the bounded-error oracle in [HMdW03]. The analysis of this strategy is tedious and can be found in Section 4 of [MNRS11], but the bottom line is that it is still possible to achieve constant-bounded error overall while repeating phase estimation only $O(1)$ times on average. In Chapter 4, we consider an even simpler strategy: we set $k = 1$, which is to not repeat phase estimation at all. We also use the same ancilla register throughout, instead of initializing a fresh one each time. In this strategy, we can no longer rely on the additive error bound to prove that the algorithm approximates Grover search. Instead we provide a careful analysis and determine whether the resulting algorithm is correct anyway.

We conclude our commentary on the MNRS quantum walk with a small digression, which is that the algorithm in Section 3.3 of [MNRS11] is not quite the same as our Algorithm 3.3.5. Our algorithm requires knowledge of ϵ, in the same way that Grover's amplitude amplification also requires knowledge of the ratio of marked elements to determine the number of iterations for which the algorithm needs to run. The technique used in the paper is the so-called *randomized Grover search*. We do not concern ourselves with this technique, because it adds complexity to the algorithm that does not contribute to the main points of the thesis. It is a straightforward modification of the original Grover search, and it is just as easily applicable to our work in Chapters 4 and 5.

3.4 Imperfect Checking

Throughout Section 3.3, we assumed that the checking oracle is perfect. That is, Step 2 of Algorithm 3.3.5 is exact, implemented by a checking operator U_C that computes f without error. Motivated by existing quantum walk-based algorithms for certain problems, we are interested in relaxing this assumption. That is, we would like to assume instead that U_C has constant bounded error. Under this assumption, the standard approach is to reduce error by running U_C $O(\log \frac{1}{\sqrt{\epsilon}})$ times on each iteration of Step 2 and take the majority vote on its output. With this approach, the cumulative error from $O(1/\sqrt{\epsilon})$ iterations will be sufficiently small that we can say that the resulting algorithm approximates Algorithm 3.3.5. This approach, used in conjunction with complex MNRS, gives rise to a total cost of $O(\mathsf{S} + \frac{1}{\sqrt{\epsilon}}(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C}\log\frac{1}{\sqrt{\epsilon}}))$. However, the $\log\frac{1}{\sqrt{\epsilon}}$ factor is undesirable. We would like to study whether there exists more efficient algorithms with cost $O(\mathsf{S} + \frac{1}{\sqrt{\epsilon}}(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C}))$, preferably in both query and time complexity. Chapters 4 and 5 will be devoted to finding such an algorithm.

4 Approach 1: MNRS Quantum Walk without Reduction of Error of Phase Estimation

We now study a slight simplification of the MNRS quantum walk algorithm described in Algorithm 3.3.5. In this simplification, Step 3 of the algorithm is implemented using Algorithm 3.3.11 with the parameter k set to 1. That is, the phase estimation algorithm is run only once per iteration. In addition, Step 3 uses the same ancilla register throughout the algorithm, as opposed to a fresh ancilla on every iteration. We analyze the behaviour of this algorithm, in the hope of discovering that when the checking procedure *is* perfect, we do not need any special tricks to reduce the error of Step 3 of Algorithm 3.3.5, either by simply running phase estimation multiple times, or by varying the number of times it is run in a complex pattern. This is so that when U_C is *not* perfect, we can avoid incurring the logarithmic majority voting overhead by treating it as a bounded error oracle and use the error reduction strategy in [HMdW03]. This approach does not assume any structure in U_C. U_C can be a generic bounded-error oracle like the one in [HMdW03]. The said algorithm can be analyzed using the theory of principal angles and vectors, because its iterate is a product of two reflections. We determine whether the resulting rotation still results in a correct algorithm that succeeds with high probability.

In this chapter, I_n denotes the identity operator that acts on the space $\mathbb{C}^n$ for every positive integer n.

4.1 Proposed Algorithm

The proposed algorithm uses two data registers, each of dimension $|X|$, and one ancilla register of dimension $O(1/\sqrt{\delta})$. The steps are basically the same as Algorithm 3.3.5 except for two differences. First, the number of times Step 2 and 3 are repeated may be different. We let T denote the number of repetitions, where T is calculated in later sections. We would like T to be in $O(1/\sqrt{\epsilon})$ so that the performance of this algorithm is asymptotically no worse than the original MNRS quantum walk. Second, as noted previously, Step 3 is a special case of Algorithm 3.3.11 with $k = 1$ that uses the *same* ancilla register throughout.

Algorithm 4.1.1 (MNRS walk without reduction of error of phase estimation)

(1) Create the state $|\pi\rangle := \sum_{x \in X} \sqrt{\pi_x}|x\rangle \sum_{y \in X} \sqrt{p_{xy}}|y\rangle$ on the two data registers. Also, initialize the ancilla register to $|0\rangle$.

Repeat Steps 2 and 3 T times

(2) Flip the phase of marked states on the first register. That is, perform the operation

$$|x\rangle|y\rangle \mapsto \begin{cases} -|x\rangle|y\rangle & \text{if } f(x) = 1, \\ |x\rangle|y\rangle & \text{if } f(x) = 0. \end{cases}$$

(3) Reflect the two registers *approximately* about the $|\pi\rangle$ state.

(4) Measure the first register. Return the measurement outcome if it is a marked state; return "not found" otherwise.

The inner workings of Step 3, which involves the phase estimation algorithm, will be relevant to our analysis. We provide the details here.

Definition 4.1.2 (Quantum Fourier transform)

The *quantum Fourier transform* of order N, which we denote as F_N, is a unitary operator with the following action.

$$F_N : |x\rangle \mapsto \sum_{j=0}^{N-1} \exp(jx \cdot 2\pi i/N)\ |j\rangle \qquad \text{for all } x \in \{0, 1, \ldots, N-1\}$$

Definition 4.1.3 (Controlled U_W)

Suppose there is a unitary operator U_W that acts on two registers of dimension $|X|$, as defined in Definition 3.3.6, the *N-wise controlled* U_W is an operation that uses an extra control register of dimension N, with the following action.

$$\text{c-}U_W : |\phi\rangle \otimes |x\rangle \mapsto (U_W)^x|\phi\rangle \otimes |x\rangle \qquad \text{for all } x \in \{0, 1, \ldots, N-1\} \text{ and } |\phi\rangle \in \mathbb{C}^{|X|} \otimes \mathbb{C}^{|X|}$$

(The control register is the *second* register.)

Algorithm 4.1.4 (Step 3 of Algorithm 4.1.1)

We have two input registers of dimensions $|X|$ and an ancilla register of dimension $O(1/\sqrt{\delta})$, as we did in Algorithm 4.1.1. Let N denote the exact dimension of the ancilla register. Step 3 of Algorithm 4.1.1 is then implemented as follows.

(1) Apply F_N to the ancilla register.

(2) Apply c-U_W to the three registers.

(3) Apply F_N^{-1} to the ancilla register.
(4) Flip the phase if the ancilla register is non-$|0\rangle$.
(5) Uncompute the first 3 steps.
(i.e., perform the inverses of (3), (2), and (1), in that order.)

Definition 4.1.5 (Phase estimation)
The first three steps of Algorithm 4.1.4 are known as the *phase estimation* algorithm, which satisfies Theorem 3.3.10. Let Φ denote the unitary operation defined by these steps:

$$\Phi := (I_{|X|} \otimes I_{|X|} \otimes F_N^{-1}) \cdot \text{c-}U_W \cdot (I_{|X|} \otimes I_{|X|} \otimes F_N).$$

Property 4.1.6 (Inverse of phase estimation)
Suppose that $|\phi\rangle \in \mathbb{C}^{|X|} \otimes \mathbb{C}^{|X|}$ and that $|\phi\rangle$ is an eigenvector of U_W.

$$\text{If } \Phi|\phi\rangle|0\rangle = |\phi\rangle \textstyle\sum_{j=0}^{N-1} \alpha_j \, |j\rangle,$$
$$\text{Then } \Phi^{-1}|\phi\rangle|0\rangle = |\phi\rangle \textstyle\sum_{j=0}^{N-1} \alpha_j^* \, |(N-j) \bmod N\rangle.$$

Proof. To prove this result, we first establish the following lemma.
Lemma 4.1.7 $\langle\phi|\langle b| \, \Phi \, |\phi\rangle|a\rangle = \langle\phi|\big\langle (b+k) \bmod N \big| \, \Phi \, |\phi\rangle \big| (a+k) \bmod N \big\rangle$
for all $a, b \in \{0, 1, \ldots, N-1\}$ and $k \in \mathbb{Z}$.
Here is one intuition for what the lemma is trying to say: when we run phase estimation, we usually initialize the ancilla state to $|0\rangle$. So consider the special case where $a = 0$. In this case, the lemma tells us that had we started with a different state such as $|k\rangle$ on the ancilla register instead of $|0\rangle$, the output of phase estimation would be "shifted" by k. One way to prove the lemma is to tediously expand both sides using the definition of Φ: if $|\phi\rangle$ has the eigenvalue $\exp(\theta i)$, then both sides expand to $\sum_{j=0}^{N-1} \exp((aj - bj) \cdot 2\pi i/N + \theta j)$. Our desired result becomes easy to prove using the lemma. By assumption, we have $\langle\phi|\langle j|\Phi|\phi\rangle|0\rangle = \alpha_j$ for all standard basis states $|j\rangle \in \mathbb{C}^N$. Therefore,

$$\alpha_j^* = (\langle\phi|\langle j|\Phi|\phi\rangle|0\rangle)^* = \langle\phi|\langle 0|\Phi^{-1}|\phi\rangle|j\rangle = \langle\phi|\langle (N-j) \bmod N|\Phi^{-1}|\phi\rangle|0\rangle,$$

where the last equality applies Lemma 4.1.7 with $a = j$, $b = 0$, and $k = N - j$. Property 4.1.6 is of interest to us because it leads to the following corollaries. □

Corollary 4.1.8 For all $|\phi\rangle \in \mathbb{C}^{|X|} \otimes \mathbb{C}^{|X|}$,

$$|\langle\phi|\langle 0|\Phi|\phi\rangle|0\rangle| = |\langle\phi|\langle 0|\Phi^{-1}|\phi\rangle|0\rangle|.$$

(More specifically, $\langle\phi|\langle 0|\Phi|\phi\rangle|0\rangle = (\langle\phi|\langle 0|\Phi^{-1}|\phi\rangle|0\rangle)^*$.)

Corollary 4.1.9 Therefore, Corollary 4.1.8 implies that Theorem 3.3.10(4) and (5) apply to the *inverse* of phase estimation as well. That is,
(1) If $|\phi\rangle$ is a 1-eigenvector of U_W (e.g. $|\pi\rangle$), then $\langle\phi|\langle 0|\Phi^{-1}|\phi\rangle|0\rangle = 1$.
(2) If $|\phi\rangle \in \text{span}\{A, B\}$ and $\langle\pi|\phi\rangle = 0$, then $|\langle\phi|\langle 0|\Phi^{-1}|\phi\rangle|0\rangle| \leq \beta$, for some small constant β.

Corollary 4.1.9 is used in Section 4.2, where we analyze the algorithm in more detail. For the time being, we note that both Steps 2 and 3 of Algorithm 4.1.1 are reflection

operators: Step 2 is clearly a reflection operator as it effects a phase flip. Step 3, whose implementation is described in Algorithm 4.1.4, also effects a phase flip up to a similarity transformation by Φ. As such, the theory of principal angles and vectors can be used to analyze the behaviour of this algorithm. We proceed with this analysis in the next section.

4.2 Principal Angles and Vectors of Proposed Algorithm

Let Π_2 and Π_3 denote projection operators onto spaces about which the reflections occur, so that the operators corresponding to Steps 2 and 3 of Algorithm 4.1.1 are respectively $(2\Pi_2 - I)$ and $(2\Pi_3 - I)$. We see that Step 2 reflects about the space spanned by those states with unmarked elements in the first register, so $\Pi_2 = \sum_{x \in X, f(x)=0} |x\rangle\langle x| \otimes I_{|X|} \otimes I_N$. Meanwhile, there are two equivalent ways of deriving Π_3. The first way is by inspecting Algorithm 4.1.4 step by step. We can derive that $\Pi_3 = \Phi^{-1} \cdot (I_{|X|} \otimes I_{|X|} \otimes |0\rangle\langle 0|) \cdot \Phi$. As for the second way, observe that Step 3 reflects about the space spanned by those initial states that gives the $|0\rangle$ phase estimate. Motivated by this observation, we make the following definitions.

Definition 4.2.1 (of "$|v_j\rangle$")
Let $|v_1\rangle, |v_2\rangle, \ldots, |v_{|X|^2}\rangle \in \mathbb{C}^{|X|} \otimes \mathbb{C}^{|X|}$ denote the $|X|^2$ orthonormal eigenvectors of U_W. In particular, let $|v_1\rangle$ denote $|\pi\rangle$, the unique 1-eigenvector within $\text{span}\{A, B\}$, which is also the initial state of the data registers of Algorithm 4.1.1. (See Definition 3.3.6 for definitions of U_W, A, and B.)

Definition 4.2.2 (of "$|\lambda_j\rangle$")
For each $j \in \{1, 2, \ldots, |X|^2\}$, let $|\lambda_j\rangle \in \mathbb{C}^N$ be the state such that $\Phi^{-1}|v_j\rangle|0\rangle = |v_j\rangle|\lambda_j\rangle$. That is, $|\lambda_j\rangle$ denotes the contents of the ancilla register of $\Phi^{-1}|v_j\rangle|0\rangle$.

We remark that $|\lambda_1\rangle = |0\rangle$, due to Corollary 4.1.9. In fact, $|\lambda_j\rangle = |0\rangle$ for *any* j such that $|v_j\rangle$ is a 1-eigenvector of U_W. It is straightforward to verify that the states that give the $|0\rangle$ phase estimate, and therefore do not pick up a phase flip from Algorithm 4.1.4, are precisely those spanned by $\{|v_j\rangle|\lambda_j\rangle\}_{j=1,\ldots,|X|^2}$. Hence, $\Pi_3 = \sum_{1 \le j \le |X|^2} |v_j\rangle|\lambda_j\rangle\langle v_j|\langle\lambda_j|$. In summary, the projection operators corresponding to Steps 2 and 3 of Algorithm 4.1.1 are as follows.

Property 4.2.3

$$\Pi_2 = \sum_{x \in X, f(x)=0} |x\rangle\langle x| \otimes I_{|X|} \otimes I_N,$$

$$\Pi_3 = \Phi^{-1} \cdot (I_{|X|} \otimes I_{|X|} \otimes |0\rangle\langle 0|) \cdot \Phi = \sum_{j=1}^{|X|^2} |v_j\rangle|\lambda_j\rangle\langle v_j|\langle\lambda_j|.$$

Section 2.6 tells us that the nonzero singular values of the product of projectors $\Pi_3\Pi_2$ reveals to us the cosines of principal angles, and that the corresponding singular vectors are the principal vectors. Given a matrix M, one method of computing its singular values and vectors is by finding the *eigen*values and vectors of MM^*. The square roots of the eigenvalues of MM^* are the singular values of M. Moreover, for each eigenvalue, its eigenvectors are the left singular vectors of M. The matrix of interest in our case is $\Pi_3\Pi_2$. Since projection matrices are Hermitian and idempotent, we find that $(\Pi_3\Pi_2)(\Pi_3\Pi_2)^* = \Pi_3\Pi_2\Pi_2^*\Pi_3^* = \Pi_3\Pi_2\Pi_3$.

Definition 4.2.4 Let $\Pi_U := \sum_{x\in X, f(x)=0} |x\rangle\langle x| \otimes I_{|X|}$.

Property 4.2.5

$$\Pi_3\Pi_2\Pi_3 = \sum_{j=1}^{|X|^2}\sum_{k=1}^{|X|^2}\langle v_j|\Pi_U|v_k\rangle \cdot \langle\lambda_j|\lambda_k\rangle \cdot \Big(|v_j\rangle|\lambda_j\rangle\langle v_k|\langle\lambda_k|\Big).$$

Proof. The proof is immediate from Property 4.2.3 and Definition 4.2.4:

$$\begin{aligned}
\Pi_3\Pi_2\Pi_3 &= \left(\sum_{j=1}^{|X|^2}|v_j\rangle|\lambda_j\rangle\langle v_j|\langle\lambda_j|\right)\left(\sum_{x\in X, f(x)=0}|x\rangle\langle x|\otimes I_{|X|}\otimes I_N\right)\left(\sum_{k=1}^{|X|^2}|v_k\rangle|\lambda_k\rangle\langle v_k|\langle\lambda_k|\right)\\
&= \sum_{j=1}^{|X|^2}\sum_{k=1}^{|X|^2}|v_j\rangle|\lambda_j\rangle\langle v_j|\langle\lambda_j|\left(\sum_{x\in X, f(x)=0}|x\rangle\langle x|\otimes I_{|X|}\otimes I_N\right)|v_k\rangle|\lambda_k\rangle\langle v_k|\langle\lambda_k|\\
&= \sum_{j=1}^{|X|^2}\sum_{k=1}^{|X|^2}|v_j\rangle|\lambda_j\rangle\left(\langle v_j|\cdot\sum_{x\in X, f(x)=0}|x\rangle\langle x|\otimes I_{|X|}\cdot|v_k\rangle\right)\Big(\langle\lambda_j|I_N|\lambda_k\rangle\Big)\langle v_k|\langle\lambda_k|\\
&= \sum_{j=1}^{|X|^2}\sum_{k=1}^{|X|^2}|v_j\rangle|\lambda_j\rangle\Big(\langle v_j|\Pi_U|v_k\rangle\Big)\Big(\langle\lambda_j|\lambda_k\rangle\Big)\langle v_j|\langle\lambda_j|\\
&= \sum_{j=1}^{|X|^2}\sum_{k=1}^{|X|^2}\langle v_j|\Pi_U|v_k\rangle\cdot\langle\lambda_j|\lambda_k\rangle\cdot\Big(|v_j\rangle|\lambda_j\rangle\langle v_j|\langle\lambda_j|\Big).
\end{aligned}$$

□

Definition 4.2.6 Let $|u_j\rangle := \Pi_U|v_j\rangle$ for all $j \in \{1, 2, \ldots, |X|^2\}$.

Corollary 4.2.7 Property 4.2.5 can be restated thus, using Definition 4.2.6:

$$\Pi_3\Pi_2\Pi_3 = \sum_{j=1}^{|X|^2}\sum_{k=1}^{|X|^2}\langle u_j|u_k\rangle\cdot\langle\lambda_j|\lambda_k\rangle\cdot\Big(|v_j\rangle|\lambda_j\rangle\langle v_k|\langle\lambda_k|\Big).$$

We would like to find the eigenvalues and vectors of $\Pi_3\Pi_2\Pi_3$. To make our computation simpler, we make a change of basis. Consider the following definition.

Definition 4.2.8 Let R be the $|X|^2 \times |X|^2$ square matrix whose elements are given by

$$r_{jk} := \langle u_j|u_k\rangle\cdot\langle\lambda_j|\lambda_k\rangle.$$

In other words,

$$R = \sum_{j=1}^{|X|^2}\sum_{k=1}^{|X|^2}\langle u_j|u_k\rangle\cdot\langle\lambda_j|\lambda_k\rangle\cdot\Big(|j\rangle\langle k|\Big).$$

By comparing Corollary 4.2.7 and Definition 4.2.8, the reader can easily verify that all eigenvalues of $\Pi_3\Pi_2\Pi_3$ whose eigenvectors are in $\text{span}\{|v_j\rangle|\lambda_j\rangle\}_{j=1,\ldots,|X|^2}$ are also eigenvalues of R. The eigenvectors of $\Pi_3\Pi_2\Pi_3$ and R corresponding to the said eigenvalues are related by a change of basis from the basis of $\{|v_j\rangle|\lambda_j\rangle\}_{j=1,\ldots,|X|^2}$ to the standard basis.

In our definition of $|v_j\rangle$, we did not impose any restrictions on the ordering of the eigenvectors $|v_j\rangle$, other than requiring $|v_1\rangle$ to denote the unique 1-eigenvector $|\pi\rangle$ of U_W in $\text{span}\{A, B\}$. Let $m = \dim \text{span}\{A, B\}$. (For an irreducible, aperiodic, and reversible walk, $m = 2|X| - 1$, because $\dim A = \dim B = |X|$, and $\dim(A \cap B) = 1$.) We now further stipulate $|v_2\rangle, \ldots, |v_m\rangle$ to be the other $m - 1$ eigenvectors of U_W within $\text{span}\{A, B\}$. The remaining eigenvectors $|v_{m+1}\rangle, \ldots, |v_{|X|^2}\rangle$ are therefore in $(\text{span}\{A, B\})^\perp$. We order the eigenvectors within $\text{span}\{A, B\}$ before those that are not, so that the matrix R can be written in a more organized form, as we see below.

Theorem 4.2.9
For each $j \in \{1, 2, \ldots, |X|^2\}$, let
$\epsilon_j := \langle u_1|u_j\rangle$,
$\beta_j := \langle\lambda_1|\lambda_j\rangle = \langle 0|\lambda_j\rangle$.
Then,

$$R = \begin{bmatrix} 1-\epsilon & \beta_2\epsilon_2 & \cdots & \beta_m\epsilon_m & 0 & \cdots & 0 \\ \beta_2^*\epsilon_2^* & & & & & & \\ \vdots & & \star & & & \star & \\ \beta_m^*\epsilon_m^* & & & & & & \\ 0 & & & & & & \\ \vdots & & \star & & & \star & \\ 0 & & & & & & \end{bmatrix}.$$

(Recall that ϵ is the ratio of marked elements, as defined in Definition 3.2.16. The "$\star$"s represent blocks of the matrix whose values we do not explicitly compute in this theorem.)

Further, the following properties are satisfied:

(1) $\beta_1 = \beta_{m+1} = \beta_{m+2} = \cdots = \beta_{|X|^2} = 1$,
$|\beta_2|, \ldots, |\beta_m| \leq \beta$, for some small constant β,
(2) $\sum_{j=2}^{m} |\epsilon_j|^2 = \epsilon \cdot (1 - \epsilon)$,
(3) $\sqrt{\sum_{j=2}^{m} |\beta_j\epsilon_j|^2} \leq \beta\sqrt{\epsilon}\sqrt{1-\epsilon}$,
(4) $\text{trace}(R) = \text{trace}(\Pi_U)$, and $\text{trace}(\Pi_U) = (\sum_{x\in X, f(x)=0} 1) \cdot |X|$,
(5) For all $j \in \{2, \ldots, m\}$,
$$|\epsilon_j|^2 \leq \epsilon \cdot (1 - \||u_j\rangle\|^2) \qquad \text{and} \qquad |\epsilon_j|^2 \leq (1-\epsilon) \cdot \||u_j\rangle\|^2.$$

Regarding Theorem 4.2.9, Parts (1), (2), and (4) are basic results regarding the elements of the matrix R. Part (3) is a corollary from (1) and (2); it provides an upper bound on the norm of the top row (and left most column) of R excluding the first element. Part (5)

provides an upper bound for elements on the top row of R (or leftmost column) provided that we know the diagonal element on the same column (or row). When $\||u_j\rangle\|^2 > \epsilon$, the first inequality of Part (5) provides an tighter bound than the second. The opposite is true when $\||u_j\rangle\|^2 < \epsilon$. Note that Part (2) already implies that $|\epsilon_j|^2 \leq \epsilon \cdot (1-\epsilon)$ for every $j \in \{2, \ldots, m\}$, but Part (5) improves on this upper bound. For every value of $\||u_j\rangle\| \in [0, 1]$, the tighter of two inequalities provides a bound that is at least as tight as implied by Part (2). In the special cases where $\||u_j\rangle\| = 0$ or 1, the tighter inequality implies that $\epsilon_j = 0$.

Before we begin the proof of Theorem 4.2.9, we note that the proof uses two simple lemmas. They are as follows.

Lemma 4.2.10 $\||u_1\rangle\| = \sqrt{1-\epsilon}$.

Lemma 4.2.11 For all $j \in \{m+1, \ldots, |X|^2\}$, $\langle u_1|u_j\rangle = 0$.

Lemma 4.2.10 is straightforward to verify: start with $|u_1\rangle = \Pi_U|v_1\rangle = \Pi_U|\pi\rangle$ using the definitions of $|u_1\rangle$ and $|v_1\rangle$, and then plug in the definitions of Π_U, $|\pi\rangle$, and ϵ from Property 4.2.5, Algorithm 4.1.1, and Definition 3.2.16. The proof of Lemma 4.2.11 is a bit longer. It is discussed at the end of this section.

We now proceed with the proof of Theorem 4.2.9.

Proof of Theorem 4.2.9. We begin by proving that the top row and leftmost column of R are indeed as stated. The elements of the top row are computed as follows.

$$r_{11} = \langle u_1|u_1\rangle\langle\lambda_1|\lambda_1\rangle = \langle u_1|u_1\rangle = \||u_1\rangle\|^2 = 1-\epsilon, \qquad \text{(by Lemma 4.2.10)}$$

$$r_{1j} = \langle u_1|u_j\rangle\langle\lambda_1|\lambda_j\rangle = \begin{cases} \beta_j\epsilon_j & \text{if } 2 \leq j \leq m, \quad \text{(by definitions of } \beta_j \text{ and } \epsilon_j) \\ 0 & \text{if } m+1 \leq j \leq |X|^2. \quad \text{(by Lemma 4.2.11, } \epsilon_j = \langle u_1|u_j\rangle = 0) \end{cases}$$

One can verify that the matrix R is a Hermitian matrix. As such, elements of the leftmost column are given by $r_{j1} = (r_{1j})^*$.

Next, we present proof for each of the stated properties.

(1) Since $|v_1\rangle = |\pi\rangle \in A \cap B$, and $|v_{m+1}\rangle, \ldots, |v_{|X|^2}\rangle \in (\text{span}\{A, B\})^\perp$, we know from Corollary 2.5.3 that $|v_1\rangle, |v_{m+1}\rangle, \ldots, |v_{|X|^2}\rangle$ are all 1-eigenvectors of U_W. Therefore, Corollary 4.1.9(1) implies that $|\lambda_j\rangle = |0\rangle$ for all $j \in \{1, m+1, m+2, \ldots, |X|^2\}$. So

$$\beta_j = \langle 0|\lambda_j\rangle = \langle 0|0\rangle = 1$$

for all $j \in \{1, m+1, m+2, \ldots, |X|^2\}$.

Similarly, since $|v_2\rangle, \ldots, |v_m\rangle$ are all eigenvectors of U_W in $\text{span}\{A, B\}$ that are orthogonal to $|\pi\rangle$, we can use Corollary 4.1.9(2) and show that

$$\beta \geq |\langle v_j|\langle 0|\Phi^{-1}|v_j\rangle|0\rangle| = |\langle v_j|\langle 0| \cdot |v_j\rangle|\lambda_j\rangle| = |\langle 0|\lambda_j\rangle| = |\beta_j|$$

for all $j \in \{2, \ldots, m\}$.

(2) Based on the fact that $\{|v_j\rangle\}_{j=1,\ldots,|X|^2}$ forms an orthonormal basis for $\mathbb{C}^{|X|} \otimes \mathbb{C}^{|X|}$,

we deduce that $\sum_{j=1}^{|X|^2} |\langle u_1|v_j\rangle|^2 = \||u_1\rangle\|^2$. By Lemma 4.2.10, we have $\||u_1\rangle\|^2 = 1-\epsilon$. Therefore

$$\begin{aligned}
1-\epsilon &= \sum_{j=1}^{|X|^2} |\langle u_1|v_j\rangle|^2 \\
&= \sum_{j=1}^{|X|^2} |\langle u_1|\Pi_U|v_j\rangle|^2 \qquad \text{(because } \langle u_1| = \langle u_1|\Pi_U\text{)} \\
&= \sum_{j=1}^{|X|^2} |\langle u_1|u_j\rangle|^2 \qquad \text{(because } \Pi_U|v_j\rangle = |u_j\rangle\text{)} \\
&= |\langle u_1|u_1\rangle|^2 + \sum_{j=2}^{|X|^2} |\langle u_1|u_j\rangle|^2 \\
&= (1-\epsilon)^2 + \sum_{j=2}^{|X|^2} |\langle u_1|u_j\rangle|^2.
\end{aligned}$$

By rearranging, we get $\sum_{j=2}^{|X|^2} |\langle u_1|u_j\rangle|^2 = 1-\epsilon-(1-\epsilon)^2 = \epsilon\cdot(1-\epsilon)$. The desired result follows immediately, because $\sum_{j=2}^{m} |\epsilon_j|^2 = \sum_{j=2}^{m} |\langle u_1|u_j\rangle|^2 = \sum_{j=2}^{|X|^2} |\langle u_1|u_j\rangle|^2$, by definition of ϵ_j and Lemma 4.2.11.

(3) The proof steps are straightforward applications of Parts (1) and (2):

$$\sqrt{\sum_{j=2}^{m} |\beta_j\epsilon_j|^2} \le \sqrt{\sum_{j=2}^{m} |\beta\epsilon_j|^2} = \beta\sqrt{\sum_{j=2}^{m} |\epsilon_j|^2} = \beta\sqrt{\epsilon}\sqrt{1-\epsilon}.$$

(4) Reading off the diagonal elements of R from Definition 4.2.8, we get

$$\text{trace}(R) = \sum_{j=1}^{|X|^2} \langle u_j|u_j\rangle\cdot\langle\lambda_j|\lambda_j\rangle = \sum_{j=1}^{|X|^2} \langle u_j|u_j\rangle = \sum_{j=1}^{|X|^2} \langle v_j|\Pi_U|v_j\rangle.$$

Because $\{|v_j\rangle\}_{j=1,\ldots,|X|^2}$ forms an orthonormal basis of $\mathbb{C}^{|X|}\otimes\mathbb{C}^{|X|}$, we have $\langle v_j|\Pi_U|v_j\rangle = \text{trace}(\Pi_U)$, which proves that $\text{trace}(R) = \text{trace}(\Pi_U)$. Π_U is a diagonal matrix in the standard basis. By counting the number of 1s on the diagonal of Π_U from its definition in Definition 4.2.4, we see that $\text{trace}(\Pi_U) = (\sum_{x\in X, f(x)=0} 1)\cdot|X|$.

(5) Let

$$\Pi_M := (I_{|X|}\otimes I_{|X|}) - \Pi_U = \sum_{x\in X, f(x)=1} |x\rangle\langle x|\otimes I_{|X|},$$

so that $\Pi_U + \Pi_M = I_{|X|}\otimes I_{|X|}$. Let $|m_j\rangle = \Pi_M|v_j\rangle$ for all $j\in\{1,2,\ldots,|X|^2\}$. Immediately we have $|v_j\rangle = |u_j\rangle + |m_j\rangle$. We can also see that for all $j\ge 2$,

$$0 = \langle v_1|v_j\rangle = \langle v_1|(\Pi_U + \Pi_M)|v_j\rangle = \langle v_1|\Pi_U|v_j\rangle + \langle v_1|\Pi_M|v_j\rangle = \langle u_1|u_j\rangle + \langle m_1|m_j\rangle.$$

So $|\langle u_1|u_j\rangle| = |\langle m_1|m_j\rangle|$. Since $|u_j\rangle$ and $|m_j\rangle$ are orthogonal, $\||u_j\rangle\|^2 + \||m_j\rangle\|^2 = 1$. Hence

$$\||m_1\rangle\| = \sqrt{1 - \|u_1\|^2} = \sqrt{\epsilon}, \text{and}$$

$$\||m_j\rangle\| = \sqrt{1 - \|u_j\|^2}.$$

The Cauchy-Schwarz inequality implies that $|\langle m_1|m_j\rangle| \leq \|m_1\| \cdot \|m_j\| = \sqrt{\epsilon}\sqrt{1 - \|u_j\|^2}$, which in turn implies that $|\langle u_1|u_j\rangle| \leq \sqrt{\epsilon}\sqrt{1 - \|u_j\|^2}$. Squaring both sides and setting $\epsilon_j = \langle u_1|u_j\rangle$ yields the first inequality.

As for the second inequality, we use the Cauchy-Schwarz inequality directly to get

$$|\langle u_1|u_j\rangle| \leq \|u_1\| \cdot \|u_j\| = \sqrt{1-\epsilon}\,\|u_j\|,$$

so $|\epsilon_j|^2 = |\langle u_1|u_j\rangle|^2 \leq (1-\epsilon) \cdot \|u_j\|^2$, as we desired. □

We are interested in finding the eigenvalues and vectors of a matrix in form of R, which satisfies properties stated in Theorem 4.2.9. These eigenvalues, the square root of which are the singular values of $\Pi_3\Pi_2$, allow us to derive the angle of rotation within each principal subspace of the operator $(2\Pi_3 - I)(2\Pi_2 - I)$. Section 4.3 comments further on the correctness and efficiency of Algorithm 4.1.1.

We conclude this section by completing the proof for Lemma 4.2.11 as promised. It is a simple corollary of the following lemma.

Lemma 4.2.12 For all $|\phi\rangle \in A$, $|\chi\rangle \in (\text{span}\{A, B\})^\perp$, $\langle\phi|\Pi_U|\chi\rangle = 0$.

Proof. Suppose that $|\chi\rangle$ is any state in $(\text{span}\{A, B\})^\perp$. From Definition 3.3.6(1), A is spanned by $\{|x\rangle \sum_{y\in X} \sqrt{p_{xy}}|y\rangle\}_{x\in X}$. For every $j \in X$, let $|\phi_j\rangle = |j\rangle \sum_{y\in X} \sqrt{p_{jy}}|y\rangle$. Observe first that $\langle\phi_j|\chi\rangle = 0$ for all $j \in X$, because $|\phi_j\rangle \in A$ and $|\chi\rangle \in (\text{span}\{A, B\})^\perp$.

We can then show that $\langle\phi_j|\Pi_U|\chi\rangle = 0$ for each $j \in X$: if $f(j) = 0$, then $\langle\phi_j|\Pi_U = \langle\phi_j|$, so $\langle\phi_j|\Pi_U|\chi\rangle = \langle\phi_j|\chi\rangle = 0$; on the other hand, if $f(j) = 1$, then $\langle\phi_j|\Pi_U = 0$, so $\langle\phi_j|\Pi_U|\chi\rangle = 0 \cdot |\chi\rangle = 0$. Since $\langle\phi_j|\Pi_U|\chi\rangle = 0$ is true for all $j \in X$, it must also be the case that $\langle\phi|\Pi_U|\chi\rangle = 0$, where $|\phi\rangle$ is any linear combination of the $|\phi_j\rangle$s. □

Lemma 4.2.11 For all $j \in \{m+1, \ldots, |X|^2\}$, $\langle u_1|u_j\rangle = 0$.

Proof. By Definition 4.2.6, $\langle u_1|u_j\rangle = \langle v_1|\Pi_U|v_j\rangle$. Since $|v_1\rangle \in A$, and $|v_j\rangle \in (\text{span}\{A, B\})^\perp$, Lemma 4.2.12 tells us that $\langle v_1|\Pi_U|v_j\rangle = 0$. □

The reader may wonder whether a result analogous to Lemma 4.2.12 also holds for the subspace B. That is, if $|\psi\rangle \in B$ and $|\chi\rangle \in (\text{span}\{A, B\})^\perp$, whether it is the case that $\langle\psi|\Pi_U|\chi\rangle = 0$. It turns out that this is not always true. In fact, it is not even true for the Markov chain shown in Example 3.2.17. Let $|\psi_2\rangle = \frac{1}{\sqrt{2}}|1\rangle|2\rangle + \frac{1}{\sqrt{2}}|3\rangle|2\rangle$, $|\psi_4\rangle = \frac{1}{\sqrt{2}}|1\rangle|4\rangle + \frac{1}{\sqrt{2}}|3\rangle|4\rangle$, and $|\chi\rangle = |1\rangle|2\rangle - |1\rangle|4\rangle - |3\rangle|2\rangle + |3\rangle|4\rangle$. Applying the definitions of A and B in Definition 3.3.6(1)(2) to Example 3.2.17, one can verify that $|\psi_2\rangle \in B$, $|\psi_4\rangle \in B$, and $|\chi\rangle \in (\text{span}\{A, B\})^\perp$, but $\langle\psi_2|\Pi_U|\chi\rangle \neq 0$ and $\langle\psi_4|\Pi_U|\chi\rangle \neq 0$. We leave it to the interested reader to work through this example.

4.3 Finding Eigenvalues and Vectors of Matrices in the Form of R

In Section 4.1, we saw that the iterate of Algorithm 4.1.1 is a product of reflection operators. To understand its behaviour, we sought the principal angles and vectors of this operator, which can be obtained from the eigenvalues and vectors of the matrix $\Pi_3\Pi_2\Pi_3$, derived in Property 4.2.5. Under a change of basis, and with a suitable ordering of basis vectors, the said matrix is related to R, defined in Definition 4.2.8, which satisfies properties stated in Theorem 4.2.9. We want to know whether the algorithm works. Going forward with the analysis, there are two points that we wish to clarify.

First, recall that the algorithm is constructed using the quantum walk operator U_W, defined in terms of a Markov Chain, and the phase estimation algorithm Φ. However, throughout our derivations so far, we did not use any Markov chain-specific properties, other than the fact that the $|v_j\rangle$s are orthogonal and $|v_1\rangle = |\pi\rangle$. We also did not scrutinize the phase estimation algorithm any more than observing that $\Phi^{-1}|\phi\rangle|0\rangle = |0\rangle$ for all 1-eigenvectors $|\phi\rangle$ of U_W, and $|\langle\phi|\langle 0|\Phi^{-1}|\phi\rangle|0\rangle| \leq \beta$ for all non-1-eigenvectors. This begs the question of whether these key properties of Markov chains and phase estimation are all the sufficient conditions we need to prove the validity of the algorithm. The research direction undertaken in this thesis entertains this idea. Consider the following definitions.

Definition 4.3.1 (Notations used Section 4.2 in more general terms)

(1) Let M, N be positive integers.
(2) Let $f : \{1, \ldots, M\} \mapsto \{0, 1\}$ be a function.
(3) Let $\Pi_U = \sum_{x\in\{1,\ldots,M\}, f(x)=0} |x\rangle\langle x|$.
(4) Let $|v_1\rangle, \ldots, |v_M\rangle$ be *any* orthonormal basis for $\mathbb{C}^M$.
(5) Let $|u_j\rangle = \Pi_U|v_j\rangle$ for all $j \in \{1, \ldots, M\}$.
(6) Let $\epsilon = 1 - \||u_1\rangle\|^2$.
(7) Let $|\lambda_1\rangle, \ldots, |\lambda_M\rangle$ be *any* states in $\mathbb{C}^N$, satisfying
  (a) $|\lambda_1\rangle = |0\rangle$, and
  (b) for all $j \in \{2, \ldots, M\}$, $|\langle 0|\lambda_j\rangle| \leq \beta$ or $\langle u_1|u_j\rangle = 0$.
(8) Let $\Pi_2 = \Pi_U \otimes I_N$.
(9) Let $\Pi_3 = \sum_{j=1}^{M} |v_j\rangle|\lambda_j\rangle\langle v_j|\langle\lambda_j|$.
(10) Let $R = \sum_{j=1}^{M}\sum_{k=1}^{M}\langle u_j|u_k\rangle \cdot \langle\lambda_j|\lambda_k\rangle \cdot (|j\rangle\langle k|)$.

Algorithm 4.3.2

Using notations defined in Definition 4.3.1, consider an algorithm that uses two registers of dimensions M and N, with the following steps.

(1) Initialize the two registers with the state $|v_1\rangle|0\rangle$.

Repeat Steps 2 and 3 T times

(2) Perform the operation $(2\Pi_2 - I)$.
(3) Perform the operation $(2\Pi_3 - I)$.

(4) Measure the first register. Return the measurement outcome if it is a marked state, return "not found" otherwise. (Again, a state x is *marked* if $f(x) = 1$.)

Notice that Algorithm 4.1.1, which we aim to analyze, is a special case of Algorithm 4.3.2, where $M = |X|^2$, $N \in O(1/\sqrt{\delta})$, the $|v_j\rangle$s are the eigenvectors of the quantum walk operator U_W, and the $|\lambda_j\rangle$s are the inverse phase estimates of $|v_j\rangle$, satisfying $\Phi^{-1}|v_j\rangle|0\rangle = |v_j\rangle|\lambda_j\rangle$. One can verify that the said inverse phase estimates, i.e., the $|\lambda_j\rangle$s defined and used in Section 4.2, satisfy Definition 4.3.1(7a) and (7b), in particular (7b): for all $j \in \{2, \ldots, m\}$, $|\langle 0|\lambda_j\rangle| \leq \beta$, and for all $j \in \{m+1, \ldots, |X|^2\}$, $\langle 0|\lambda_j\rangle = 1$, but $\langle u_1|u_j\rangle = 0$.

Theorem 4.3.3 Analogues of Property 4.2.5, Corollary 4.2.7 and parts of Theorem 4.2.9 also apply to Π_2, Π_3 and R that were defined in Definition 4.3.1:

$$\begin{aligned}\Pi_3\Pi_2\Pi_3 &= \sum_{j=1}^{M}\sum_{k=1}^{M} |v_j\rangle\Pi_U|v_k\rangle \cdot \langle\lambda_j|\lambda_k\rangle \cdot \Big(|v_j\rangle|\lambda_j\rangle\langle v_j|\langle\lambda_j|\Big) \\ &= \sum_{j=1}^{M}\sum_{k=1}^{M} \langle u_j|u_k\rangle \cdot \langle\lambda_j|\lambda_k\rangle \cdot \Big(|v_j\rangle|\lambda_j\rangle\langle v_j|\langle\lambda_j|\Big).\end{aligned}$$

If we let $\epsilon_j := \langle u_1|u_j\rangle$, and $\beta_j := \langle\lambda_1|\lambda_j\rangle = \langle 0|\lambda_j\rangle$, then the following are true.

(∗) $r_{11} = 1 - \epsilon$,
(1) $\beta_1 = 1$,
(2) $\sum_{j=2}^{M} |\epsilon_j|^2 = \epsilon \cdot (1 - \epsilon)$,
(3) $\sqrt{\sum_{j=2}^{M} |\beta_j\epsilon_j|^2} \leq \beta\sqrt{\epsilon}\sqrt{1-\epsilon}$,
(4) $\text{trace}(R) = \text{trace}(\Pi_U)$,
(5) For all $j \in \{2, \ldots, M\}$,

$$|\epsilon_j|^2 \leq \epsilon \cdot (1 - \||u_j\rangle\|^2) \qquad \text{and} \qquad |\epsilon_j|^2 \leq (1-\epsilon) \cdot \||u_j\rangle\|^2.$$

We omit the proof for Theorem 4.3.3, since it follows basically the same steps as the analogous properties in Section 4.2.

The next point of clarification is to define what we mean by "the algorithm works". What criteria regarding the principal angles and vectors must be satisfied for us to consider the algorithm to be successful? Recall that the goal of the algorithm is to *find* a marked state *quickly*. So correctness and efficiency are two criteria the algorithm needs to satisfy. Formally, what we mean the following.

Problem 4.3.4 We wish to determine whether Algorithm 4.1.1 is *correct* and *efficient*.

(1) The algorithm is *correct* if there exists T such that, after T iterations, the probability of measuring a marked element is at least a constant.

(2) The algorithm is *efficient* if $T \in O(1/\sqrt{\epsilon})$. Otherwise the original MNRS algorithm is more efficient.

(3) We attempt to determine the correctness and efficiency of Algorithm 4.3.2 that uses the more general definitions of $|v_j\rangle$ and $|\lambda_j\rangle$ in Definition 4.3.1 that are agnostic of Markov

chains and phase estimation, because then the correctness and efficiency of Algorithm 4.1.1, being a special case of Algorithm 4.3.2, will be implied.

The number of iterations T that the algorithm requires depends on the principal angles of the iterate. To facilitate further discussion, we make the following definitions.

Definition 4.3.5
Let $M_2, M_3 \subseteq \mathbb{C}^M \otimes \mathbb{C}^N$ be subspaces that Π_2 and Π_3 projects onto.
Let ℓ be the smaller one of $\dim M_2$ and $\dim M_3$.

Let $\theta_1, \theta_2, \ldots, \theta_\ell$ be the principal angles between M_2 and M_3.
Let $|s_1\rangle, |s_2\rangle, \ldots, |s_\ell\rangle \in M_2$ and $|t_1\rangle, |t_2\rangle, \ldots, |t_\ell\rangle \in M_3$ be the principal vectors.

We note that the initial state of the algorithm $|v_1\rangle|0\rangle$ is in M_3. Also, the action of $(2\Pi_3 - I)(2\Pi_2 - I)$ within *every* principal subspace is a Grover-like rotation towards a marked state. Using language from Chapter 2: for every $j \in \{1, \ldots, \ell\}$, within $\text{span}\{|s_j\rangle, |t_j\rangle\}$, $(2\Pi_3 - I)(2\Pi_2 - I)$ effects a rotation by $2\theta_j$ from $|s_j\rangle$ to $|s_j^\perp\rangle$, where by Gram-Schmidt, $|s_j^\perp\rangle = |t_j\rangle - |s_j\rangle\langle s_j|t_j\rangle$, normalized. But $|s_j^\perp\rangle$ contains only marked elements, because it is precisely $|t_j\rangle$ with unmarked elements subtracted away.

So one can imagine a few ways in which the algorithm may be correct. For example, perhaps the initial state $|v_1\rangle|0\rangle$ has a large projection onto one particular principal subspace, and the principal angle of this subspace is in $O(1/\sqrt{\epsilon})$. Or perhaps $|v_1\rangle|0\rangle$ has large projections onto several principal subspaces, and all the principal angles are in $O(1/\sqrt{\epsilon})$ and are close to each other.

Problem 4.3.6
Is Algorithm 4.3.2 correct and efficient for all $|v_j\rangle$ and $|\lambda_j\rangle$ satisfying Definition 4.3.1?

Unfortunately, the short answer to the above problem is a disappointing "no". Despite all the properties we were able to prove in Theorem 4.2.9 and Theorem 4.3.3, we could explicitly find the eigenvalues and vectors of R only for specific simple cases. Section 4.4 presents such a simple case: the case where $|\lambda_2\rangle = \cdots = |\lambda_M\rangle$. Even so, we were able to produce a counterexample where Algorithm 4.3.2 is not correct. Section 4.5 presents the counterexample. It is currently uncertain whether the counterexample can be circumvented.

Before moving on, we would like to state and prove one property regarding principal angles that is used in both Sections 4.4 and 4.5. This property allows us to find, in simple cases, how much the initial state $|v_1\rangle|0\rangle$ projects onto the principal subspaces without explicitly computing the principal vectors.

Property 4.3.7 (Weighted average of eigenvalues of R)
We use the same notations as we did in Definition 4.3.5. Let $k = \dim M_3$. If $k > \ell$, let $|t_{\ell+1}\rangle, \ldots, |t_k\rangle$ be states such that $|t_1\rangle, \ldots, |t_k\rangle$ form an orthonormal basis for M_3.

$$\text{If } |v_1\rangle|0\rangle = \sum_{j=1}^{k} \alpha_j |t_j\rangle, \quad \text{then } \sum_{j=1}^{k} |\alpha_j|^2 \cos^2\theta_j = 1 - \epsilon.$$

Explanation. Since $|v_1\rangle|0\rangle \in M_3$, it can be written as a weighted sum (i.e., linear combination) of the orthonormal basis states $|t_j\rangle$. The property says that the *weighted average* of eigenvalues of R, which are the squares of singular values $\cos\theta_j$ of $\Pi_3\Pi_2$, is $1-\epsilon$.

Proof.

$$\begin{aligned}
1-\epsilon &= \langle u_1|\langle 0| \cdot |u_1\rangle|0\rangle \\
&= \langle v_1|\langle 0| \Pi_2 |v_1\rangle|0\rangle \\
&= \left(\sum_{i=1}^{k} \alpha_i^* \langle t_i|\right) \Pi_2 \left(\sum_{j=1}^{k} \alpha_j |t_j\rangle\right) \\
&= \sum_{i=1}^{k}\sum_{j=1}^{k} \alpha_i^* \alpha_j \langle t_i|\Pi_2|t_j\rangle \\
&= \sum_{i=1}^{k}\sum_{j=1}^{k} \alpha_i^* \alpha_j \cos\theta_j \langle t_i|s_j\rangle \qquad \big(\text{by Property 2.2.4, } \Pi_2|t_j\rangle = \cos\theta_j|s_j\rangle\big) \\
&= \sum_{j=1}^{k} \alpha_j^* \alpha_j \cos^2\theta_j \qquad \left(\text{by Property 2.2.2, } \langle t_i|s_j\rangle = \begin{cases} \cos\theta_j & \text{if } i=j \\ 0 & \text{if } i\neq j \end{cases}\right) \\
&= \sum_{j=1}^{k} |\alpha_j|^2 \cos^2\theta_j
\end{aligned}$$

□

4.4 The Simple Case

We now present the analysis of a special case of Algorithm 4.3.2. In this special case, we suppose that $|\lambda_1\rangle = 0$, and $|\lambda_2\rangle = \cdots = |\lambda_M\rangle$, where $\langle 0|\lambda_2\rangle = \cdots = \langle 0|\lambda_M\rangle = \beta$ for some small constant β. Hypothetically, one can imagine the scenario where the quantum walk operator U_W has just a single 1-eigenvector $|v_1\rangle$, and all the other eigenvectors $|v_2\rangle, \ldots, |v_M\rangle$ have the same eigenvalue and therefore the same inverse phase estimates $|\lambda_2\rangle, \ldots, |\lambda_M\rangle$. We say "hypothetically", because we remark at the end of this section that this case will almost never happen for a non-trivial quantum walk algorithm. Nonetheless, we present the analysis for this case because it is a valid special case for Algorithm 4.3.2 which we are able to completely analyze from start to finish.

Theorem 4.4.1 When $|\lambda_2\rangle = \cdots = |\lambda_M\rangle$, Algorithm 4.3.2 is correct and efficient. If $\langle 0|\lambda_2\rangle = \beta$, then for small values of $\epsilon > 0$ and small values of β, a marked element can be discovered with high probability by running Algorithm 4.3.2 for $O\big(1/\sqrt{\epsilon\,(1-|\beta|^2)}\big)$ iterations.

In the following analysis, we use notations that were defined in Section 4.3, particularly in Definition 4.3.1 and Definition 4.3.5. As we derived previously, the behaviour of Algorithm 4.3.2 is characterized by the eigenvalues and vectors of $\Pi_3\Pi_2\Pi_3$, whose formula is stated in Theorem 4.3.3. $\Pi_3\Pi_2\Pi_3$ is then related to the matrix R, defined in Definition 4.3.1(10).

Definition 4.4.2 Let $|\lambda\rangle = |\lambda_2\rangle$.
(Since $|\lambda_2\rangle, \ldots, |\lambda_M\rangle$ are all equal, we drop the redundant subscript from our notation.)

Definition 4.4.3 Let $\Pi_M = I_M - \Pi_U$ and $|m_1\rangle = \Pi_M|v_1\rangle$, so that $|v_1\rangle = |u_1\rangle + |m_1\rangle$.

Definition 4.4.4 Let $|\hat{u}_1\rangle = |u_1\rangle/\||u_1\|$ and $|\hat{m}_1\rangle = |m_1\rangle/\||m_1\|$.

Property 4.4.5 Since $\||u_1\rangle\| = \sqrt{1-\epsilon}$ and $\||m_1\rangle\| = \sqrt{\epsilon}$, we have

$$|v_1\rangle = \sqrt{1-\epsilon}\,|\hat{u}_1\rangle + \sqrt{\epsilon}\,|\hat{m}_1\rangle.$$

From the definition of Π_3, we see that the subspace M_3 is spanned by $\{|v_j\rangle|\lambda_j\rangle\}_{j=1,\ldots,M}$. In our case, $\{|v_j\rangle|\lambda_j\rangle\}_{j=1,\ldots,M} = \{|v_1\rangle|0\rangle\} \cup \{|v_j\rangle|\lambda\rangle\}_{j=2,\ldots,M}$. Since $\{|v_j\rangle\}_{j=1,\ldots,M}$ forms an orthonormal basis for $\mathbb{C}^M$, any definition of $|v_2\rangle, \ldots, |v_M\rangle$ such that $\{|v_j\rangle\}_{j=2,\ldots,M}$ forms an orthonormal basis for the orthogonal complement of $|v_1\rangle$ in $\mathbb{C}^M$ would be consistent with the definition of Π_3 and leads to a correct analysis. We choose to use the following definition of $|v_2\rangle, \ldots, |v_M\rangle$, because the analysis it leads to has the simplest steps.

Definition 4.4.6 Let $|v_2\rangle = \sqrt{\epsilon}\,|\hat{u}_1\rangle - \sqrt{1-\epsilon}\,|\hat{m}_1\rangle$. One can confirm that $\langle v_1|v_2\rangle = 0$. Let $|v_3\rangle, \ldots, |v_M\rangle$ be any states such that $|v_1\rangle, \ldots, |v_M\rangle$ form an orthonormal basis for $\mathbb{C}^M$.

Property 4.4.7 $\langle u_j|u_k\rangle = 0$ for all $j \in \{1,2\}$ and $k \in \{3, \ldots, M\}$.

Proof. We first prove the following lemma: for all $k \in \{3, \ldots, M\}$, $\langle \hat{u}_1|v_k\rangle = 0$. Since $\{|v_k\rangle\}_{k=1,\ldots,M}$ forms an orthonormal basis for $\mathbb{C}^M$, we have $\sum_{k=1}^{M} |\langle \hat{u}_1|v_k\rangle|^2 = \||\hat{u}_1\rangle\|^2 = 1$. Hence,

$$
\begin{aligned}
1 &= \sum_{k=1}^{M} |\langle \hat{u}_1 | v_k \rangle|^2 \\
&= |\langle \hat{u}_1 | v_1 \rangle|^2 + |\langle \hat{u}_1 | v_2 \rangle|^2 + \sum_{k=3}^{M} |\langle \hat{u}_1 | v_k \rangle|^2 \\
&= (1-\epsilon) + \epsilon + \sum_{k=3}^{M} |\langle \hat{u}_1 | v_k \rangle|^2 \\
&= 1 + \sum_{k=3}^{M} |\langle \hat{u}_1 | v_k \rangle|^2 \\
0 &= \sum_{k=3}^{M} |\langle \hat{u}_1 | v_k \rangle|^2.
\end{aligned}
$$

So $|\langle \hat{u}_1 | v_k \rangle|^2 = 0$ for all $k \in \{3, \ldots, M\}$, implying that $\langle \hat{u}_1 | v_k \rangle = 0$. The lemma has been proven. Then, using the lemma, we can say that for all $k \in \{3, \ldots, M\}$,

$$
\begin{aligned}
&\langle u_1 | u_k \rangle = \langle v_1 | \Pi_U | v_k \rangle = \sqrt{1-\epsilon}\, \langle \hat{u}_1 | v_k \rangle = 0, \text{and} \\
&\langle u_2 | u_k \rangle = \langle v_2 | \Pi_U | v_k \rangle = \sqrt{\epsilon}\, \langle \hat{u}_1 | v_k \rangle = 0.
\end{aligned}
$$

□

Property 4.4.8 Let $\beta = \langle 0 | \lambda \rangle$, then

$$
R = \begin{bmatrix}
1-\epsilon & \beta\sqrt{\epsilon}\sqrt{1-\epsilon} & 0 & \cdots & 0 \\
\beta^*\sqrt{\epsilon}\sqrt{1-\epsilon} & \epsilon & 0 & \cdots & 0 \\
0 & 0 & & & \\
\vdots & \vdots & & \star & \\
0 & 0 & & &
\end{bmatrix}.
$$

Proof. Definition 4.3.1(10) tell us that elements of R are given by $r_{jk} = \langle u_j | u_k \rangle \cdot \langle \lambda_j | \lambda_k \rangle$. From the definitions of $|v_1\rangle$ and $|v_2\rangle$, we know that $|u_1\rangle = \Pi_U |v_1\rangle = \sqrt{1-\epsilon}\, |\hat{u}_1\rangle$ and $|u_2\rangle = \Pi_U |v_2\rangle = \sqrt{\epsilon}\, |\hat{u}_1\rangle$. From here, we compute that

$$
\begin{aligned}
r_{11} &= \langle u_1 | u_1 \rangle \cdot \langle \lambda_1 | \lambda_1 \rangle = 1 - \epsilon\, \langle \hat{u}_1 | \hat{u}_1 \rangle \langle 0 | 0 \rangle = 1 - \epsilon, \\
r_{22} &= \langle u_2 | u_2 \rangle \cdot \langle \lambda_2 | \lambda_2 \rangle = \epsilon\, \langle \hat{u}_1 | \hat{u}_1 \rangle \langle \lambda | \lambda \rangle = \epsilon, \\
r_{12} &= \langle u_1 | u_2 \rangle \cdot \langle \lambda_1 | \lambda_2 \rangle = \sqrt{1-\epsilon}\sqrt{\epsilon}\, \langle \hat{u}_1 | \hat{u}_1 \rangle \langle 0 | \lambda \rangle = \beta\sqrt{\epsilon}\sqrt{1-\epsilon}, \\
r_{21} &= (r_{12})^* = \beta^*\sqrt{\epsilon}\sqrt{1-\epsilon}.
\end{aligned}
$$

Further, for all $k \in \{3, \ldots, M\}$, Property 4.4.7 tells us that

$$
\begin{aligned}
r_{1k} &= \langle u_1|u_k\rangle \cdot \langle \lambda_1|\lambda_k\rangle = 0 \cdot \langle \lambda_1|\lambda_k\rangle = 0, \\
r_{2k} &= \langle u_2|u_k\rangle \cdot \langle \lambda_2|\lambda_k\rangle = 0 \cdot \langle \lambda_2|\lambda_k\rangle = 0, \\
r_{k1} &= (r_{1k})^* = 0, \\
r_{k2} &= (r_{2k})^* = 0.
\end{aligned}
$$

□

We see that, under our choice of basis, the R matrix is block diagonal. In our analysis, we need only to concern ourselves with the top left 2×2 block, because the eigenvectors of $\Pi_3\Pi_2\Pi_3$ corresponding to the bottom right block are orthogonal to the initial state $|v_1\rangle|0\rangle$.

Property 4.4.9 The two eigenvalues of the matrix

$$
\begin{bmatrix} 1-\epsilon & \beta\sqrt{\epsilon}\sqrt{1-\epsilon} \\ \beta^*\sqrt{\epsilon}\sqrt{1-\epsilon} & \epsilon \end{bmatrix}
$$

are $\lambda_1 = \dfrac{1}{2} + \sqrt{\dfrac{1}{4} - \epsilon \cdot (1-\epsilon) \cdot \left(1 - |\beta|^2\right)}$ and $\lambda_2 = \dfrac{1}{2} - \sqrt{\dfrac{1}{4} - \epsilon \cdot (1-\epsilon) \cdot \left(1 - |\beta|^2\right)}$.

Proof. The characteristic equation of the matrix is $\lambda^2 - \lambda + \epsilon \cdot (1-\epsilon) \cdot (1 - |\beta|^2) = 0$. Solving this equation for λ using the quadratic formula gives the desired result. □

We remark that the two eigenvalues stated by Property 4.4.9 can be simplified when $|\beta| = 0$ or 1. When $|\beta| = 1$, $\frac{1}{2} \pm \sqrt{\frac{1}{4} - \epsilon \cdot (1-\epsilon) \cdot \left(1 - |\beta|^2\right)} = \frac{1}{2} \pm \sqrt{\frac{1}{4}} = 0$ or 1. When $|\beta| = 0$, $\frac{1}{2} \pm \sqrt{\frac{1}{4} - \epsilon \cdot (1-\epsilon) \cdot \left(1 - |\beta|^2\right)} = \frac{1}{2} \pm \sqrt{\left(\frac{1}{2} - \epsilon\right)^2} = \frac{1}{2} \pm \left|\frac{1}{2} - \epsilon\right| = \epsilon$ or $1 - \epsilon$. However, we are most interested in the case where β is a small constant close to zero. For that, we have the following approximations.

Property 4.4.10 For all $\epsilon \in [0, \frac{1}{2})$,

$$
\lambda_1 \leq 1 - \epsilon + \frac{1-\epsilon}{1-2\epsilon}\,\epsilon\,|\beta|^2 \qquad \text{and}
$$

$$
\lambda_2 \geq \epsilon - \frac{1-\epsilon}{1-2\epsilon}\,\epsilon\,|\beta|^2.
$$

Proof. We present the proof for λ_1. For all $\epsilon \in [0, \frac{1}{2})$,

$$
\begin{aligned}
\lambda_1 &= \frac{1}{2} + \sqrt{\frac{1}{4} - \epsilon\cdot(1-\epsilon)\cdot\left(1-|\beta|^2\right)} \\
&= \frac{1}{2} + \sqrt{\frac{1}{4} - \epsilon\cdot(1-\epsilon) + |\beta|^2\cdot\epsilon\cdot(1-\epsilon)} \\
&= \frac{1}{2} + \sqrt{\frac{1}{4} - \epsilon\cdot(1-\epsilon)}\sqrt{1 + \frac{|\beta|^2\cdot\epsilon\cdot(1-\epsilon)}{\frac{1}{4} - \epsilon\cdot(1-\epsilon)}} \\
&= \frac{1}{2} + \sqrt{\left(\frac{1}{2}-\epsilon\right)^2}\sqrt{1 + \frac{|\beta|^2\cdot\epsilon\cdot(1-\epsilon)}{\frac{1}{4} - \epsilon\cdot(1-\epsilon)}} \\
&= \frac{1}{2} + \left(\frac{1}{2}-\epsilon\right)\sqrt{1 + \frac{|\beta|^2\cdot\epsilon\cdot(1-\epsilon)}{\frac{1}{4} - \epsilon\cdot(1-\epsilon)}} \\
&\leq \frac{1}{2} + \left(\frac{1}{2}-\epsilon\right)\left(1 + \frac{1}{2}\cdot\frac{|\beta|^2\cdot\epsilon\cdot(1-\epsilon)}{\frac{1}{4} - \epsilon\cdot(1-\epsilon)}\right) \qquad \left(\text{because } \sqrt{1+x} \leq 1 + \tfrac{1}{2}x\right) \\
&= \frac{1}{2} + \left(\frac{1}{2}-\epsilon\right) + \left(\frac{1}{2}-\epsilon\right)\cdot\frac{1}{2}\cdot\frac{|\beta|^2\cdot\epsilon\cdot(1-\epsilon)}{\frac{1}{4} - \epsilon\cdot(1-\epsilon)} \\
&= 1 - \epsilon + \left(\frac{1}{2}-\epsilon\right)\cdot\frac{1}{2}\cdot\frac{|\beta|^2\cdot\epsilon\cdot(1-\epsilon)}{\frac{1}{4} - \epsilon\cdot(1-\epsilon)} \\
&= 1 - \epsilon + \left(\frac{1}{2}-\epsilon\right)\cdot\frac{1}{2}\cdot\frac{|\beta|^2\cdot\epsilon\cdot(1-\epsilon)}{\left(\frac{1}{2}-\epsilon\right)^2} \\
&= 1 - \epsilon + \frac{1-\epsilon}{1-2\epsilon}\cdot\epsilon\cdot|\beta|^2
\end{aligned}
$$

The proof for λ_2 is similar, except the first "+" signs in the first six lines are replaced with "−"s, the "≤" is replaced with "≥", and we simplify accordingly afterwards. □

The key observation from Property 4.4.10 is that for small values of ϵ, $\frac{1-\epsilon}{1-2\epsilon} \approx 1$, so for small values of β, the eigenvalue λ_1 is close to $1-\epsilon$, because $\lambda_1 \leq 1 - \epsilon + \frac{1-\epsilon}{1-2\epsilon}\,\epsilon\,|\beta|^2 \approx 1 - \epsilon + \epsilon\,|\beta|^2$. It turns out, as we prove below, that the initial state $|v_1\rangle|0\rangle$ also has a large projection onto the eigenvector of λ_1.

Property 4.4.11 Let $|t_1\rangle$, $|t_2\rangle$ be eigenvectors of $\Pi_3\Pi_2\Pi_3$ with eigenvalues λ_1, λ_2. Then $|\langle t_1|\cdot|v_1\rangle|0\rangle|^2 \geq 1-\epsilon$ for all $\epsilon \in [0, \frac{1}{2})$ and all β such that $|\beta| \in [0,1]$.

Proof. We first note that $\lambda_1 \in [1-\epsilon, 1]$ for all $|\beta| \in [0,1]$: as $|\beta|$ increases from 0 to 1, $\lambda_1 = \frac{1}{2} + \sqrt{\frac{1}{4} - \epsilon\cdot(1-\epsilon)\cdot\left(1-|\beta|^2\right)}$ increases monotonically from $1-\epsilon$ to 1.
Let $\rho_1 = |\langle t_1|\cdot|v_1\rangle|0\rangle|^2$ and $\rho_2 = |\langle t_2|\cdot|v_1\rangle|0\rangle|^2$. The fact that R is block diagonal means that eigenvectors of $\Pi_3\Pi_2\Pi_3$ other than $|t_1\rangle$ and $|t_2\rangle$ are all orthogonal to $|v_1\rangle|0\rangle$. So by Property 4.3.7,

$$\rho_1\lambda_1 + \rho_2\lambda_2 = 1 - \epsilon.$$

Then, since $\rho_1 + \rho_2 = 1$ and $\lambda_1 + \lambda_2 = 1$,

$$\rho_1\lambda_1 + (1-\rho_1)(1-\lambda_1) = 1 - \epsilon.$$

Solving for ρ_1 gives us

$$\rho_1 = \frac{\epsilon - \frac{1}{2}}{1 - 2\lambda_1} + \frac{1}{2}.$$

One can then show from here that ρ_1 decreases monotonically from 1 to $1-\epsilon$ as λ_1 increases from $1-\epsilon$ to 1. Note that the numerator $\epsilon - \frac{1}{2}$ is negative because $\epsilon \in [0, \frac{1}{2})$. □

Corollary 4.4.12 By running Algorithm 4.3.2 for $O\big(1/\sqrt{\epsilon\,(1-|\beta|^2)}\big)$ iterations, the resulting state is marked with high probability.

Proof. The proof of Corollary 4.4.12, which we now present, provides a summary of our findings in this section. Since the iterate of Algorithm 4.3.2 is a product of two reflections, its behaviour can be understood using the theory of principal angles and vectors. The eigenvectors of $\Pi_3\Pi_2\Pi_3$, which are the left singular vectors of $\Pi_3\Pi_2$, give us one of the two principal vectors in each principal subspace, namely $|t_1\rangle, |t_2\rangle, \ldots, |t_\ell\rangle$. Since R, the matrix related to $\Pi_3\Pi_2\Pi_3$, is block diagonal, we know that the initial state $|v_1\rangle|0\rangle$ has nonzero projection onto only two principal subspaces: $\mathrm{span}\{|s_1\rangle, |t_1\rangle\}$ and $\mathrm{span}\{|s_2\rangle, |t_2\rangle\}$. ($|s_1\rangle := \Pi_2|t_1\rangle$ normalized, $|s_2\rangle := \Pi_2|t_1\rangle$ normalized.) Of these two subspaces, $|v_1\rangle|0\rangle$ has a large projection onto $\mathrm{span}\{|s_1\rangle, |t_1\rangle\}$ in particular: by Property 4.4.11, the square of the norm of this projection is at least $1-\epsilon$. With every iteration of the algorithm, the component of $|v_1\rangle|0\rangle$ within this principal subspace rotates towards a marked state by angle $2\theta_1$, where θ_1 is the principal angle of this subspace. The said principal angle satisfies $\cos^2\theta_1 = \lambda_1$. By Property 4.4.10, λ_1 is bounded away from 1. To be formal, we consider ϵ to be *small* if $\epsilon \in (0, \frac{1}{10}]$. For such small values of ϵ, $\frac{1-\epsilon}{1-2\epsilon} \in (1, \frac{9}{8}]$. We have

$$\begin{aligned}
\theta_1 &= \cos^{-1}\sqrt{\lambda_1} \\
&= \sin^{-1}\sqrt{1-\lambda_1} \\
&\geq \sin^{-1}\sqrt{\epsilon - \tfrac{1-\epsilon}{1-2\epsilon}\,\epsilon\,|\beta|^2} \qquad \text{(by Property 4.4.10)} \\
&\geq \sin^{-1}\sqrt{\epsilon - \tfrac{9}{8}\,\epsilon\,|\beta|^2} \\
&\geq \sqrt{\epsilon\,(1 - \tfrac{9}{8}|\beta|^2)} \\
&\in \Omega\left(\sqrt{\epsilon\,(1-|\beta|^2)}\right) \text{ for small values of } \beta.
\end{aligned}$$

Because $|v_1\rangle|0\rangle$ has a large projection onto a principal subspace whose principal angle is in $\Omega\big(\sqrt{\epsilon\,(1-|\beta|^2)}\big)$, we deduce that running the algorithm for $O\big(1/\sqrt{\epsilon\,(1-|\beta|^2)}\big)$ iterations results in a marked state with high probability. Therefore, in the special case where $|\lambda_2\rangle = \cdots = |\lambda_M\rangle$, Algorithm 4.3.2 is correct and efficient. □

While the success for this special case is good news, there are two reasons why this case will most likely not occur for a quantum walk algorithm of interest. First, by inspection of Corollary 2.5.3, the eigenvalues of product of reflections that are not 1 or -1 always come in pairs: if $\exp(\theta i)$ is an eigenvalue, so is $\exp(-\theta i)$. This goes contrary to the fact that $|\lambda_2\rangle = \cdots = |\lambda_M\rangle$, which implies that there is only *one* non-1 eigenvalue, Second, $|v_1\rangle$ is generally not be the only 1-eigenvector of U_W; vectors in $(\text{span}\{A, B\}^{\perp})$ are also 1-eigenvectors. This brings us to the question of whether Algorithm 4.3.2 is correct and efficient in all cases. As we see in the next section, it is not.

4.5 The Counterexample

A question that naturally arises from Property 4.3.7 is whether this property alone is a sufficient condition for the efficiency of Algorithm 4.3.2. After all, if the weighted *average* eigenvalues of $\Pi_3\Pi_2\Pi_3$ is $1-\epsilon$, where the weights are the squared norms of projections from $|v_1\rangle|0\rangle$ to the eigenvectors, then perhaps the *individual* eigenvalues are also somewhat close to $1-\epsilon$, which would then imply the efficiency of the algorithm. In this section, we prove that this is *not* the case.

Theorem 4.5.1 There exists $\{|v_j\rangle\}_{j=1,\ldots,M}$ and $\{|\lambda_j\rangle\}_{j=1,\ldots,M}$ satisfying Definition 4.3.1 such that when $\epsilon \ll |\beta|^2$, Algorithm 4.3.2 is not correct.

The condition $\epsilon \ll |\beta|^2$ is satisfied for most search problems of interest. Typically, ϵ is very small. Meanwhile, β, the "error" of phase estimation, is a constant independent from ϵ. In fact, we require β to be an independent constant in order to prove Theorem 3.3.10(3), which says that the cost of phase estimation is in $O(\frac{1}{\sqrt{\delta}}\mathsf{U})$.

We build up this counterexample step by step. To begin, we point out a basic linear algebra fact. Consider a block matrix of the form

$$\begin{bmatrix} a & b\cdot\hat{u}^* \\ b^*\cdot\hat{u} & c\cdot I \end{bmatrix}$$

where a, b, c, are scalar constants, $\hat{u}$ is a unit column vector with, say m components, and I is the $m\times m$ identity matrix. Compare it with the 2×2 matrix

$$\begin{bmatrix} a & b \\ b^* & c \end{bmatrix}.$$

It is perhaps not a surprise that the two eigenvalues of the 2×2 matrix are also eigenvalues of the $(m+1)\times(m+1)$ block matrix.

Property 4.5.2 The eigenvalues of

$$\begin{bmatrix} a & b\cdot\hat{u}^* \\ b^*\cdot\hat{u} & c\cdot I \end{bmatrix}$$

are $\lambda_1, \lambda_2, c, \ldots, c$ if and only if the eigenvalues of

$$\begin{bmatrix} a & b \\ b^* & c \end{bmatrix}$$

are λ_1, λ_2.

Proof. Because the latter matrix is Hermitian, it has two orthonormal eigenvectors v_1, v_2 with eigenvalues λ_1, λ_2. If $v_1 = \begin{bmatrix} p \\ q \end{bmatrix}$ and $v_2 = \begin{bmatrix} r \\ s \end{bmatrix}$, one can verify that $\begin{bmatrix} p \\ q\cdot\hat{u} \end{bmatrix}$ and $\begin{bmatrix} r \\ s\cdot\hat{u} \end{bmatrix}$ are eigenvectors of the former matrix with the same respective eigenvalues. The other $m-1$ eigenvectors of the former matrix are in the form $\begin{bmatrix} 0 \\ \hat{u}^\perp \end{bmatrix}$, where $\hat{u}^\perp$ is any vector orthogonal to $\hat{u}$. These vectors have the eigenvalue c. □

Having established Property 4.5.2, we turn our attention to the following matrix.

Definition 4.5.3

$$\bar{R} := \begin{bmatrix} 1-\epsilon & \beta\sqrt{\epsilon}\sqrt{1-\epsilon} \\ \beta^*\sqrt{\epsilon}\sqrt{1-\epsilon} & 1-|\beta|^2+|\beta|^2\epsilon \end{bmatrix}$$

We will explain why this matrix is important in a moment. For the time being, we compute its eigenvalues.

Property 4.5.4 The eigenvalues of $\bar{R}$ are 1 and $(1-|\beta|^2)(1-\epsilon)$.

Proof. The characteristic equation of the matrix is $\lambda^2 - (1+(1-\epsilon)(1-|\beta|^2))\lambda + (1-\epsilon)(1-|\beta|^2) = 0$. Factoring the polynomial gives us its two solutions. □

As we derived in Sections 4.2 and 4.3, the behaviour of Algorithm 4.3.2 is related to the eigenvalues and vectors of the R matrix. If it is possible to construct $\{|v_j\rangle\}_{j=1,\ldots,M}$ and $\{|\lambda_j\rangle\}_{j=1,\ldots,M}$ such that $R = \bar{R}$, then we would have successfully disproven the correctness of the algorithm. This is because one of the eigenvalues of $\bar{R}$ is 1. The principal subspace corresponding to the eigenvalue 1 has a principal angle of zero, so no rotation happens here. We will see in Property 4.5.15 that $|v_1\rangle|0\rangle$ has a very large projection onto this principal subspace. However, it is impossible for $R = \bar{R}$, because $\bar{R}$ violates Theorem 4.3.3(5). Using notations defined in Section 4.3, the top right element of $\bar{R}$ is equal to $\beta_2\epsilon_2$. Theorem 4.3.3(5) says that $|\epsilon_2|^2 \leq \epsilon\,(1-\|u_2\|^2)$. But in our case of $\bar{R}$, we have $|\epsilon_2|^2 = \epsilon\cdot(1-\epsilon)$ and $\epsilon\,(1-\|u_2\|^2) = \epsilon\cdot(|\beta|^2 - |\beta|^2\epsilon) = |\beta|^2\cdot\epsilon\cdot(1-\epsilon)$. The left hand side is greater than the right.

Nonetheless, in the special case where $1/|\beta|^2$ is a power of 2, it *is* possible to find $\{|v_j\rangle\}_{j=1,\ldots,M}$ and $\{|\lambda_j\rangle\}_{j=1,\ldots,M}$, where $M = 1 + 1/|\beta|^2$, such that

$$R = \begin{bmatrix} 1-\epsilon & \beta^2\sqrt{\epsilon}\sqrt{1-\epsilon} & \cdots & \beta^2\sqrt{\epsilon}\sqrt{1-\epsilon} \\ (\beta^*)^2\sqrt{\epsilon}\sqrt{1-\epsilon} & 1-|\beta|^2+|\beta|^2\epsilon & & \\ \vdots & & \ddots & \\ (\beta^*)^2\sqrt{\epsilon}\sqrt{1-\epsilon} & & & 1-|\beta|^2+|\beta|^2\epsilon \end{bmatrix}.$$

$$\begin{matrix} 1 & & 1/|\beta|^2 & \end{matrix}$$

(The numbers below the matrix represent dimensions of the blocks above them.)

This matrix R is related to $\bar{R}$ through Property 4.5.2. Because of this, eigenvalues of $\bar{R}$ are also eigenvalues of R, and the eigenvectors are related by a change in basis. In addition, the matrix R satisfies Theorem 4.3.3(5), because the elements on the top row, namely $\beta^2\sqrt{\epsilon}\sqrt{1-\epsilon}$, are smaller by a factor β compared to the top right element of $\bar{R}$, which is $\beta\sqrt{\epsilon}\sqrt{1-\epsilon}$. Because the eigenvalues and vectors of $\bar{R}$ does not imply correctness of Algorithm 4.3.2, neither does R, which completes the proof.

The rest of this section explains the construction of $\{|v_j\rangle\}_{j=1,\ldots,M}$ and $\{|\lambda_j\rangle\}_{j=1,\ldots,M}$ in detail. Our construction uses the well-known Hadamard transformation.

Definition 4.5.5 (Hadamard transformation)

$$\text{Let } H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Property 4.5.6 Let "$\cdot$" denote the binary bitwise dot product, then

$$H^{\otimes k}|x\rangle = \frac{1}{\sqrt{2^k}} \sum_{y=0}^{2^k-1} (-1)^{x \cdot y} |y\rangle \qquad \text{for all } x \in \{0, \ldots, 2^k - 1\}.$$

The key property of $H^{\otimes k}$ relevant to our purpose is the following.

Corollary 4.5.7

$$\left| \langle y|H^{\otimes k}|x\rangle \right| = \left| \frac{1}{\sqrt{2^k}} \right| \qquad \text{for all } x, y \in \{0, \ldots, 2^k - 1\}.$$

$$\text{In particular, } \langle y|H^{\otimes k}|x\rangle = \frac{1}{\sqrt{2^k}} \qquad \text{if } x = 0 \text{ or } y = 0.$$

Said in another way, all elements in the Hadamard matrix have the same magnitude, namely $1/\sqrt{2^k}$; only the signs are different. Moreover, the top row and leftmost column of the matrix are positive.

We are now ready to provide the construction for our counterexample, using notations from Definition 4.3.1. Suppose that $1/|\beta|^2$ is a power of 2.

Definition 4.5.8 Let k be the positive integer such that $2^k = 1/|\beta|^2$.

Definition 4.5.9 (of M, N, f)

(1) Suppose that $M \gg 2^k + 1$. Let $N = 2^k + 1$.
(Recall the meanings of M and N from Definition 4.3.1(4) and (7).)

(2) Let $f : \{1, \ldots, M\} \mapsto \{0, 1\}$ be a function such that there is only 1 marked element.
(So that there are $M - 1$ unmarked elements.)

Definition 4.5.10 (of "$|v_j\rangle$")

Suppose that $|v_j\rangle \in \mathbb{C}^M$ for all $j \in \{1, \ldots, M\}$.

(1) Let $|v_1\rangle = \sqrt{1-\epsilon}\,|\hat{u}_1\rangle + \sqrt{\epsilon}\,|\hat{m}_1\rangle$, where
$|\hat{u}_1\rangle$ is any normalized superposition of unmarked states, and $|\hat{m}_1\rangle$ is the marked state.

(2) Let $|v_2\rangle, \ldots, |v_{2^k+1}\rangle$ be defined as follows.

$$[v_2; v_3; \ldots; v_{2^k+1}] := [w; q_1; \ldots; q_{2^k-1}] \cdot H^{\otimes k}, \text{ where}$$

(2a) $|w\rangle := \sqrt{\epsilon}\,|\hat{u}_1\rangle - \sqrt{1-\epsilon}\,|m_1\rangle$, and
(2b) $|q_1\rangle, \ldots, |q_{M-2}\rangle$ forms an orthonormal basis for the image of $\Pi_U - |\hat{u}_1\rangle\langle\hat{u}_1|$.

$$\text{(e.g. } |v_2\rangle = \tfrac{1}{\sqrt{2^k}}(|w\rangle + |q_1\rangle + \cdots + |q_{2^k-1}\rangle))$$

(3) Let $|v_{2^k+2}\rangle, \ldots, |v_M\rangle$ be defined as follows.

$$|v_j\rangle := |q_{j-2}\rangle \qquad \text{for all } j \in \{2^k + 2, \ldots, M\}$$

Definition 4.5.11 (of "$|\lambda_j\rangle$")
Suppose that $|\lambda_j\rangle \in \mathbb{C}^N = \text{span}\{|0\rangle, |1\rangle, \ldots, |2^k\rangle\}$ for all $j \in \{1, \ldots, M\}$.
(1) Let $|\lambda_1\rangle = |0\rangle$.
(2) Let $|\lambda_2\rangle, \ldots, |\lambda_{2^k+1}\rangle$ be defined as follows.

$$[\lambda_2; \lambda_3; \ldots; \lambda_{2^k+1}] := \begin{bmatrix} H^{\otimes k} \\ 0 \end{bmatrix}$$

$$(\text{e.g. } |\lambda_2\rangle = \tfrac{1}{\sqrt{2^k}}(|0\rangle + |1\rangle + \cdots + |2^k - 1\rangle))$$

(3) Let $|\lambda_{2^k+2}\rangle, \ldots, |\lambda_M\rangle$ be defined as follows.

$$|\lambda_j\rangle := |2^k\rangle \qquad \text{for all } j \in \{2^k + 2, \ldots, M\}.$$

The claim is that $\{|v_j\rangle\}_{j=1,\ldots,M}$ and $\{|\lambda_j\rangle\}_{j=1,\ldots,M}$, as defined above, gives us the counterexample.

Property 4.5.12
(1) $\langle u_1|u_1\rangle = 1 - \epsilon$
(2) $\langle u_1|u_j\rangle = \begin{cases} \beta\sqrt{1-\epsilon}\sqrt{\epsilon} & \text{if } 2 \le j \le 2^k + 1 \\ 0 & \text{if } j > 2^k + 1 \end{cases}$
(3) $\langle u_j|u_j\rangle = \begin{cases} 1 - |\beta|^2 + |\beta|^2\epsilon & \text{if } 2 \le j \le 2^k + 1 \\ 1 & \text{if } j > 2^k + 1 \end{cases}$
(4) $\langle u_i|u_j\rangle = 0 \quad \text{if } i \ne j \text{ and } 2 \le i \text{ and } 2 \le j$
(5) $\langle \lambda_j|\lambda_j\rangle = 1 \quad \text{for all } j \in \{1, \ldots, M\}$
(6) $\langle \lambda_1|\lambda_j\rangle = \begin{cases} \frac{1}{\sqrt{2^k}} = \beta & \text{if } 2 \le j \le 2^k + 1 \\ 0 & \text{if } j > 2^k + 1 \end{cases}$
(7) $\langle \lambda_i|\lambda_j\rangle = \begin{cases} 0 & \text{if } i \ne j \text{ and } 2 \le i \le 2^k + 1 \text{ and } 2 \le j \le 2^k + 1 \\ 0 & \text{if } i \ne j \text{ and } 2 \le i \le 2^k + 1 \text{ and } j > 2^k + 1 \\ 0 & \text{if } i \ne j \text{ and } i > 2^k + 1 \text{ and } 2 \le j \le 2^k + 1 \\ 1 & \text{if } i \ne j \text{ and } i > 2^k + 1 \text{ and } j > 2^k + 1 \end{cases}$

Proof. Recall that $|u_j\rangle = \Pi_U|v_j\rangle$. The proofs of all of the above are trivial from Definition 4.5.10, Definition 4.5.11, and Corollary 4.5.7.
(1) Immediate from Definition 4.3.1(6).
(2) Let $[\langle u_1|u_2\rangle; \langle u_1|u_3\rangle; \ldots; \langle u_1|u_{2^k+1}\rangle]$ denote the 1×2^k matrix whose elements are $\langle u_1|u_2\rangle, \langle u_1|u_3\rangle, \ldots, \langle u_1|u_{2^k+1}\rangle$. Then

$$
\begin{aligned}
&[\langle u_1|u_2\rangle; \langle u_1|u_3\rangle; \ldots; \langle u_1|u_{2^k+1}\rangle] \\
&= \langle v_1|\,\Pi_U\,[v_2; v_3; \ldots; v_{2^k+1}] \\
&= \langle v_1|\,\Pi_U\,[w; q_1; \ldots; q_{2^k-1}] \cdot H^{\otimes k} \\
&= \sqrt{1-\epsilon}\,\langle \hat{u}_1| \cdot [\sqrt{\epsilon}\,\hat{u}_1; q_1; \ldots; q_{2^k-1}] \cdot H^{\otimes k} \\
&= [\sqrt{1-\epsilon}\sqrt{\epsilon}; 0; \ldots; 0] \cdot H^{\otimes k} \\
&= \left[\frac{1}{\sqrt{2^k}}\sqrt{1-\epsilon}\sqrt{\epsilon}; \ldots; \frac{1}{\sqrt{2^k}}\sqrt{1-\epsilon}\sqrt{\epsilon}\right] \\
&= [\beta\sqrt{1-\epsilon}\sqrt{\epsilon}; \ldots; \beta\sqrt{1-\epsilon}\sqrt{\epsilon}].
\end{aligned}
$$

For all $j > 2^k + 1$,

$$\langle u_1|u_j\rangle = \langle v_1|\Pi_U|v_j\rangle = \sqrt{1-\epsilon}\,\langle \hat{u}_1|q_{j-2}\rangle = 0.$$

(3) For all $j \in \{2, \ldots 2^k + 1\}$,

$$
\begin{aligned}
\langle u_j|u_j\rangle &= \||u_j\rangle\|^2 \\
&= \|\Pi_U|v_j\rangle\|^2 \\
&= \left\| \frac{1}{\sqrt{2^k}}\big(\Pi_U|w\rangle \pm \Pi_U|q_1\rangle \pm \cdots \pm \Pi_U|q_{2^k-1}\rangle\big) \right\|^2 \qquad \begin{array}{c}\text{(by Definition 4.5.10(2)} \\ \text{and Corollary 4.5.7)}\end{array} \\
&= \left\| \frac{1}{\sqrt{2^k}}\big(\sqrt{\epsilon}\,|\hat{u}_1\rangle \pm |q_1\rangle \pm \cdots \pm |q_{2^k-1}\rangle\big) \right\|^2 \\
&= \frac{1}{2^k}(\epsilon + 1 + \cdots + 1) \\
&= \frac{\epsilon + 2^k - 1}{2^k} \\
&= 1 - |\beta|^2 + |\beta|^2\epsilon.
\end{aligned}
$$

For all $j > 2^k + 1$,

$$\langle u_j|u_j\rangle = \langle q_{j-2}|\Pi_U|q_{j-2}\rangle = \langle q_{j-2}|q_{j-2}\rangle = 1.$$

(4) First, note that $\Pi_U|w\rangle = \sqrt{\epsilon}\,|\hat{u}_1\rangle$, $\Pi_U|q_j\rangle = |q_j\rangle$ for all $j \in \{1, \ldots, M-2\}$, and $|\hat{u}_1\rangle, |q_1\rangle, \ldots, |q_{M-2}\rangle$ are orthogonal. Even though by Definition 4.5.10(2), the vectors $|u_2\rangle, \ldots, |u_{2^k+1}\rangle$ are defined by

$$
\begin{aligned}
[u_2; \ldots; u_{2^k+1}] &= \Pi_U\,[v_2; \ldots; v_{2^k+1}] \\
&= \Pi_U\,[w; q_1; \ldots; q_{2^k-1}] \cdot H^{\otimes k} = [\sqrt{\epsilon}\,\hat{u}_1; q_1; \ldots; q_{2^k-1}] \cdot H^{\otimes k},
\end{aligned}
$$

the columns of $[\sqrt{\epsilon}\,\hat{u}_1; q_1; \ldots; q_{2^k-1}] \cdot H^{\otimes k}$ are still orthogonal, because $H^{\otimes k}$ is a unitary transformation.

(5) Immediate from Definition 4.5.11; $|\lambda_j\rangle$ is normalized for all $j \in \{1, \ldots, M\}$.

(6) Let $[\langle\lambda_1|\lambda_2\rangle; \ldots; \langle\lambda_1|\lambda_{2^k+1}\rangle]$ denote the 1×2^k matrix whose elements are $\langle\lambda_1|\lambda_2\rangle, \ldots,$ $\langle\lambda_1|\lambda_{2^k+1}\rangle$. Then

$$
\begin{aligned}
&[\langle\lambda_1|\lambda_2\rangle; \ldots; \langle\lambda_1|\lambda_{2^k+1}\rangle] \\
&= \langle\lambda_1| \cdot [\lambda_2; \ldots; \lambda_{2^k+1}] \\
&= \langle 0| \cdot \begin{bmatrix} H^{\otimes k} \\ 0 \end{bmatrix} \\
&= [1; 0; \ldots; 0] \cdot \begin{bmatrix} H^{\otimes k} \\ 0 \end{bmatrix} \\
&= \left[\frac{1}{\sqrt{2^k}}; \ldots; \frac{1}{\sqrt{2^k}}\right] \\
&= [\beta; \ldots; \beta].
\end{aligned}
$$

For all $j > 2^k + 1$, $\langle\lambda_1|\lambda_j\rangle = \langle 0|2^k\rangle = 0$.

(7) $|\lambda_2\rangle, |\lambda_3\rangle, \ldots, |\lambda_{2^k+1}\rangle$ are all orthogonal due to their definitions in Definition 4.5.11(2): all columns of $H^{\otimes k}$ are orthogonal. These states are also orthogonal to $|2^k\rangle$. Consequently they are orthogonal to $|\lambda_{2^k+2}\rangle, \ldots, |\lambda_M\rangle$. The inner product between any two of $|\lambda_{2^k+2}\rangle, \ldots, |\lambda_M\rangle$ is 1, because they are all equal to $|2^k\rangle$. □

As a result of Property 4.5.12, we derive that the matrix R takes on the following form.

Corollary 4.5.13

$$
R = \begin{bmatrix}
1-\epsilon & \beta^2\sqrt{\epsilon}\sqrt{1-\epsilon} & \cdots & \beta^2\sqrt{\epsilon}\sqrt{1-\epsilon} & 0 & \cdots & 0 \\
(\beta^*)^2\sqrt{\epsilon}\sqrt{1-\epsilon} & 1-|\beta|^2+|\beta|^2\epsilon & & & 0 & & \\
\vdots & & \ddots & & & \ddots & \\
(\beta^*)^2\sqrt{\epsilon}\sqrt{1-\epsilon} & & & 1-|\beta|^2+|\beta|^2\epsilon & & & 0 \\
0 & 0 & & & 1 & & \\
\vdots & & \ddots & & & \ddots & \\
0 & & & 0 & & & 1
\end{bmatrix}
$$

1 (column block) $2^k\ (=1/|\beta|^2)$ (column block) $M-1-2^k$ (column block)

(Again, the numbers below the matrix represent dimensions of the blocks above them.)

Proof. Recall that according to Definition 4.3.1(10), $R = \sum_{i=1}^{M}\sum_{j=1}^{M}\langle u_i|u_j\rangle \cdot \langle\lambda_i|\lambda_j\rangle \cdot (|i\rangle\langle j|)$. Property 4.5.12 provides us with enough information to compute every element of this matrix. We leave it to the reader to work through the steps. □

Corollary 4.5.14 The eigenvalues of R are

$$1, (1-|\beta|^2)(1-\epsilon), 1-|\beta|^2+|\beta|^2\epsilon, \ldots, 1-|\beta|^2+|\beta|^2\epsilon, 1, \ldots, 1.$$

Proof. We first look at the bottom right $(M-1-2^k)\times(M-1-2^k)$ block of R. This is a diagonal matrix with 1s on the diagonal, so the eigenvalues within this block are $1, \ldots, 1$. Next, we examine the top left $(1+2^k)\times(1+2^k)$ block. Observe that this block is in the form of the block matrix stated in Property 4.5.2, with $a = 1-\epsilon$, $b = \beta\sqrt{\epsilon}\sqrt{1-\epsilon}$, and

$c = 1 - |\beta|^2 + |\beta|^2\epsilon$. As a result, two of the eigenvalues of this block are the same as those of $\bar{R}$, defined in Definition 4.5.3. These two eigenvalues are 1 and $(1 - |\beta|^2)(1 - \epsilon)$, as calculated in Property 4.5.4. By Property 4.5.2, the other $2^k - 1$ eigenvalues in this block are all equal to c, which is $1 - |\beta|^2 + |\beta|^2$ in our case. □

Let $\lambda_1 = 1$, $\lambda_2 = (1 - |\beta|^2)(1 - \epsilon)$. We then have the following.

Property 4.5.15 Suppose that $\epsilon \neq 0$, $\epsilon \neq 1$, and $|\beta|^2 \neq 0$. Let $|t_1\rangle$, $|t_2\rangle$ be the two eigenvectors of $\Pi_3\Pi_2\Pi_3$ that are not orthogonal to $|v_1\rangle|0\rangle$ with eigenvalues λ_1, λ_2. Then

$$|\langle t_1| \cdot |v_1\rangle|0\rangle|^2 = 1 - \frac{\epsilon}{|\beta|^2 + \epsilon(1 - |\beta|^2)} \quad \text{and} \quad |\langle t_2| \cdot |v_1\rangle|0\rangle|^2 = \frac{\epsilon}{|\beta|^2 + \epsilon(1 - |\beta|^2)}.$$

Proof. The state $|v_1\rangle|0\rangle$ is orthogonal to all eigenvectors of $\Pi_3\Pi_2\Pi_3$ except those corresponding to the two eigenvalues of $\bar{R}$, calculated in Property 4.5.4. Consequently, we can find the squared norms of projections of $|v_1\rangle|0\rangle$ to the two eigenvectors using Property 4.3.7. Let $\rho_1 = |\langle t_1| \cdot |v_1\rangle|0\rangle|^2$ and $\rho_2 = |\langle t_2| \cdot |v_1\rangle|0\rangle|^2$. Property 4.3.7 tells us that

$$\rho_1\lambda_1 + \rho_2\lambda_2 = 1 - \epsilon.$$

Using the fact that $\rho_1 + \rho_2 = 1$, $\lambda_1 = 1$, and $\lambda_2 = (1 - |\beta|^2)(1 - \epsilon)$. We have

$$(1 - \rho_2) \cdot 1 + \rho_2(1 - |\beta|^2)(1 - \epsilon) = 1 - \epsilon.$$

Solving for ρ_2 gives us

$$\rho_2 = \frac{\epsilon}{|\beta|^2 + \epsilon(1 - |\beta|^2)}.$$

Then ρ_1 is simply $1 - \rho_2$. □

Corollary 4.5.16 When $\epsilon \ll |\beta|^2$, Algorithm 4.3.2 is *not* correct for our choice of $\{|v_j\rangle\}_{j=1,\ldots,M}$ and $\{|\lambda_j\rangle\}_{j=1,\ldots,M}$.

Proof. Recall from Problem 4.3.4 what it means for Algorithm 4.3.2 to be correct: it is correct if the initial state $|v_1\rangle|0\rangle$ eventually rotates to a mostly marked state after a certain number of iterations. We now explain why this does not happen in our case. As always, we reason about the behaviour of the algorithm using the theory of principal angles and vectors. According to Property 4.5.15, the initial state $|v_1\rangle|0\rangle$ has a large projection onto the 1-eigenvector $|t_1\rangle$ of $\Pi_3\Pi_2\Pi_3$. Because $|t_1\rangle$ is a 1-eigenvector, there are two things we can say about the principal subspace containing it. First, no rotation happens in this subspace. The rotation angle, $2\theta_1$, is given by $2\theta_1 = 2\cos^{-1}\sqrt{1} = 0$. Second, $|t_1\rangle$ contains only unmarked states. This is because by Property 2.2.4, $1 = \cos^2\theta_1 = \|\Pi_2|t_1\rangle\|^2$, and $\|\Pi_2|t_1\rangle\|^2$ gives us the ratio of unmarked elements of $|t_1\rangle$. (See Definition 4.3.1(8) for the definition of Π_2.) Let $|\phi_T\rangle$ denote the state of Algorithm 4.3.2 after T iterations. Since no rotation happens in the principal subspace containing $|t_1\rangle$, we deduce that $|\langle t_1|\phi_T\rangle|^2$ remains equal no matter how many iterations we run the algorithm for. Hence, we derive from Property 4.5.15 that

$$|\langle t_1|\phi_T\rangle|^2 = 1 - \frac{\epsilon}{|\beta|^2 + \epsilon(1 - |\beta|^2)}$$

for *every* T. Observe that $1 - \epsilon/(|\beta|^2 + \epsilon(1 - |\beta|^2)) \geq 1 - \epsilon/|\beta|^2$. Since $|t_1\rangle$ is entirely unmarked, the probability of measuring a marked element in $|\phi_T\rangle$ therefore does not exceed $\epsilon/|\beta|^2$. So the success probability is no greater than $\epsilon/|\beta|^2$ for every T. If we assume that $\epsilon \ll |\beta|^2$ and consider $|\beta|^2$ to be a constant independent from ϵ, then the probability is in $O(\epsilon)$. Since $\epsilon = 1/M$, this probability is arbitrarily close to zero for sufficiently large values of M. Therefore, for our choice of $\{|v_j\rangle\}_{j=1,\ldots,M}$ and $\{|\lambda_j\rangle\}_{j=1,\ldots,M}$, the algorithm is *not* correct. □

4.6 Perturbation Bounds

In our analysis of Algorithm 4.3.2 so far, we determined the behaviour of the algorithm by computing the eigenvalues of $\Pi_3\Pi_2\Pi_3$ exactly. However, the process of computing eigenvalues is not trivial, and we were only able to do so in special cases such as those presented in Sections 4.4 and 4.5. Perhaps this leaves the reader wondering whether there are approximation theorems in linear algebra that are applicable to our analysis. In short, the attempted efforts of using approximation theorems has not been met with much success in the course of our research. But for completeness we highlight here some of our key methodologies.

We present one approach that one might take to prove the correctness of Algorithm 4.1.1 without explicitly calculating the eigenvalues of R. Recall that the R matrix, to which $\Pi_3\Pi_2\Pi_3$ is related, has the following form.

$$R = \begin{bmatrix} 1-\epsilon & \beta_2\epsilon_2 & \cdots & \beta_m\epsilon_m & 0 & \cdots & 0 \\ \beta_2^*\epsilon_2^* & & & & & & \\ \vdots & & \star & & & \star & \\ \beta_m^*\epsilon_m^* & & & & & & \\ 0 & & & & & & \\ \vdots & & \star & & & \star & \\ 0 & & & & & & \end{bmatrix}.$$

(See Theorem 4.2.9 for details on this matrix.) The question of whether Algorithm 4.1.1 is correct arises from the fact that phase estimation is not exact. However, if we suppose for a moment that phase estimation *is* exact—for example, suppose that all the eigenvalues of U_W are of the form $\exp(2\pi i \cdot 0.x_1x_2\ldots x_N)$ for some finite N, where $x_1, \ldots, x_N$ are binary digits—then the algorithm would be correct. In the case where phase estimation is exact, $\beta_2, \ldots, \beta_m$ are all zero. This motivates us to decompose the R matrix as

$$R = \begin{bmatrix} 1-\epsilon & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & & & & & & \\ \vdots & & \star & & & \star & \\ 0 & & & & & & \\ 0 & & & & & & \\ \vdots & & \star & & & \star & \\ 0 & & & & & & \end{bmatrix} + \begin{bmatrix} 0 & \beta_2\epsilon_2 & \cdots & \beta_m\epsilon_m & 0 & \cdots & 0 \\ \beta_2^*\epsilon_2^* & & & & & & \\ \vdots & & 0 & & & 0 & \\ \beta_m^*\epsilon_m^* & & & & & & \\ 0 & & & & & & \\ \vdots & & 0 & & & 0 & \\ 0 & & & & & & \end{bmatrix},$$

where the first term is the *ideal* form of the R matrix had there been no error in phase estimation, while the second term is the *error*, or *perturbation*, from the ideal. If the R matrix were just the first term, the proof of correctness of Algorithm 4.1.1 is easy: R is block diagonal, and the eigenvector of $\Pi_3\Pi_2\Pi_3$ corresponding to the top left 1×1 block is precisely $|v_1\rangle|0\rangle$ with eigenvalue $1-\epsilon$. The question is how much the perturbation term

affects the eigenvalue $1-\epsilon$ and its associated eigenvector $|v_1\rangle|0\rangle$. To answer this question, we use the following theorem

Theorem 4.6.1 (Perturbation theorem, Chapter 6 of [Bha13], page 152)
Let A, B be $n \times n$ Hermitian matrices. Suppose that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ are eigenvalues of A, and $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_n$ are eigenvalues of $A+B$. Then

$$|\lambda_j - \mu_j| \leq \|B\| \qquad \text{for all } j \in \{1, \ldots, n\},$$

where $\|B\|$ is the spectral norm of B.

Let A and B be the first and second term of R respectively. We know that one of the eigenvalues of A is $1-\epsilon$. Suppose that this is the jth largest eigenvalue of A, so $\lambda_j = 1-\epsilon$. It is not difficult to derive that the spectral norm of B is the norm of its top row or leftmost column. Therefore, from Theorem 4.2.9(3), $\|B\| \leq \beta\sqrt{\epsilon}\sqrt{1-\epsilon}$. Theorem 4.6.1 then tells us that the jth largest eigenvalue of $A+B$ is within $[1-\epsilon-\beta\sqrt{\epsilon}\sqrt{1-\epsilon}, 1-\epsilon+\beta\sqrt{\epsilon}\sqrt{1-\epsilon}]$. However, this is not useful to us at all. We would like this eigenvalue to be close to $1-\epsilon$, but $\sqrt{\epsilon}$ is far too large in comparison to ϵ. So this particular line of reasoning does not seem to show any promise.

The above approach is not the only method of approximating the eigenvalues of $\Pi_3\Pi_2\Pi_3$. Other means of placing bounds on eigenvalues include the theory of Gershgorin discs (See Chaper 6 [HJ12]) and the theory of sum of Hermitian matrices (See [KT01]). It is also possible to examine the matrix $\Pi_2\Pi_3\Pi_2$, because its eigenvalues are the same as those of $\Pi_3\Pi_2\Pi_3$, which are the squares of singular values of $\Pi_3\Pi_2$.

The approximation of eigenvectors presents its own set of challenges, because in general eigenvectors can be very sensitive to perturbations. Chapter 7 of [Bha13] provides the following example. Consider the two matrices $\begin{bmatrix} 1+\epsilon & 0 \\ 0 & 1-\epsilon \end{bmatrix}$ and $\begin{bmatrix} 1 & \epsilon \\ \epsilon & 1 \end{bmatrix}$. Even though they differ by only a small amount, one can easily verify that $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ are eigenvectors of the first matrix, while $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$ are eigenvectors of the second matrix.

In the end, we abandoned the approach of approximating the eigenvalues of $\Pi_3\Pi_2\Pi_3$. This is because the goal of the approximation is to prove the correctness of Algorithm 4.3.2 by reasoning that despite the small perturbation, the resulting algorithm still resembles the ideal case where the initial state $|v_1\rangle|0\rangle$ has an eigenvalue of $1-\epsilon$. However, we know that this is *not* the case, because we produced a counterexample in Section 4.5.

5 Approach 2: Composed Walk Construction

Our last topic of discussion involves the more specific problem of nested MNRS quantum walks. The problem under consideration is still Markov chain search: we are finding a marked vertex on a Markov chain on some state space, say $\{1, \ldots, n\}$, with transition matrix P. However, the process of determining whether a vertex $x \in X$ is marked requires yet another quantum walk with its own transition matrix P_x. Unlike in Chapter 4, we do assume structure in the checking procedure U_C: we assume that U_C is another MNRS quantum walk. So in a sense, we want to find a way to efficiently *compose* the outer walk on P with the inner subwalks on $P_1, \ldots, P_n$. We make the assumption that a vertex x in the outer walk is marked iff a there exists a marked vertex in the subwalk P_x. This assumption is applicable to some problems such as Element Distinctness and Triangle Finding, but not others such as AND-OR tree evaluation. (To determine whether an AND-OR tree evaluates to 0, we need to search for an OR subtree *without* a marked element.) We study a construction where we build a composed Markov chain from P and $P_1, \ldots, P_n$ and comment on its performance.

In this chapter, the (j, k)-element in matrix P is denoted as p_{jk}. The (j, k)-element in matrices $P_1, P_2, \ldots, P_n$ are denoted as $p_{jk}^{(1)}, p_{jk}^{(2)}, \ldots, p_{jk}^{(n)}$. We will also use the Dirac bra-ket notation to not only denote quantum states but vectors in general, because it provides us a visually appealing way of distinguishing row and column vectors. We let $|1\rangle, |2\rangle, \ldots, |n\rangle$ denote the standard basis column vectors in $\mathbb{C}^n$. For every vector $|\phi\rangle \in \mathbb{C}^n$, $\langle\phi|$ is a row vector that is the conjugate transpose of $|\phi\rangle$.

5.1 Composition of Markov chains

Let P denote the matrix that characterizes the "outer" Markov chain. We will refer to vertices in this Markov chain as the "outer vertices." Assuming that the state space of this Markov chain is of size n, P is an $n \times n$ matrix:

$$P = \begin{bmatrix} p_{11} & \cdots & p_{1n} \\ \vdots & \ddots & \vdots \\ p_{n1} & \cdots & p_{nn} \end{bmatrix}.$$

If the said Markov chain is irreducible, let $\langle s|$ denote the unique 1-left-eigenvector of this matrix, normalized so that its elements sum to 1. (This is the same as the π vector, defined in Definition 3.2.4.) Let s_j denote the jth element of $\langle s|$. So $\langle s| = \sum_{1 \leq j \leq n} s_j \langle j|$.

Let P_1, P_2, ..., P_n denote the matrices characterizing the "inner" Markov chains. Vertices in these Markov chains are referred to as "inner vertices." Let m_1, m_2, ..., m_n denote the sizes of their state spaces.

$$P_1 = \begin{bmatrix} p^{(1)}_{11} & \cdots & p^{(1)}_{1,m_1} \\ \vdots & \ddots & \vdots \\ p^{(1)}_{m_1,1} & \cdots & p^{(1)}_{m_1,m_1} \end{bmatrix}, \quad \cdots \quad , P_n = \begin{bmatrix} p^{(n)}_{11} & \cdots & p^{(n)}_{1,m_n} \\ \vdots & \ddots & \vdots \\ p^{(n)}_{m_n,1} & \cdots & p^{(n)}_{m_n,m_n} \end{bmatrix}.$$

If these Markov chains are irreducible, let $\langle s^{(1)}|$, $\langle s^{(2)}|$, ..., $\langle s^{(n)}|$ denote their unique 1-left-eigenvectors, normalized so that elements from each vector sum to 1.

Let $|u\rangle$ denote the column vector of 1s: $|u\rangle = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$.

Then, the composed Markov chain is defined as follows.

Definition 5.1.1 (Markov chain composed from $P, P_1, \ldots, P_n$)
Let P_{cmp} denote the matrix that characterizes the Markov chain that composes P with $P_1, \ldots, P_n$. We define this matrix to be the following.

$$P_{\text{cmp}} := \begin{bmatrix} p_{11} P_1 & p_{12} |u\rangle\langle s^{(2)}| & \cdots & p_{1n} |u\rangle\langle s^{(n)}| \\ p_{21} |u\rangle\langle s^{(1)}| & p_{22} P_2 & & p_{2n} |u\rangle\langle s^{(n)}| \\ \vdots & & \ddots & \vdots \\ p_{n1} |u\rangle\langle s^{(1)}| & p_{n2} |u\rangle\langle s^{(2)}| & \cdots & p_{nn} P_n \end{bmatrix}$$

Observe that each "element" in the above definition is a block matrix. The block on the jth row and kth column is of dimension $m_j \times m_k$. Consequently, we index the rows and columns of P_{cmp} by *two* indices: the first index is the block number, while the second index is the row or column number within the block. For example, $p^{\text{cmp}}_{(1,2),(1,3)}$ refers to the $(2,3)$-element in the $(1,1)$-block of P_{cmp}. So $p^{\text{cmp}}_{(1,2),(1,3)} = (p_{11}P_1)_{23} = p_{11}p^{(1)}_{23}$. Also note that the dimension of $|u\rangle$ is derived from the size of the block containing it.

5.2 Properties of P_{cmp}

Property 5.2.1 If $P, P_1, \ldots, P_n$ are stochastic, irreducible, aperiodic, and reversible, so is P_{cmp}.

Proof. We divide the proof into four parts. Each parts proves one of the four claimed properties of P_{cmp}.

(1) P_{cmp} is stochastic.

By assumption, $P, P_1, \ldots, P_n$ are all stochastic. Hence all rows in these matrices sum to 1. It is also the case that elements of each of $\langle s^{(1)}|$, $\langle s^{(2)}|$, ..., $\langle s^{(n)}|$ sum to 1 as well. It is straightforward to confirm that all rows of P_{cmp} sum to 1 as a result.

(2) P_{cmp} is irreducible.

To prove irreducibility, we need to show that there exists a path from any vertex to any vertex. Let $(u_{\text{src}}, v_{\text{src}})$ and $(u_{\text{dest}}, v_{\text{dest}})$ be two vertices in P_{cmp}. (As noted in Definition 5.1.1, every vertex in P_{cmp} labelled by two indices. The first label is the index of the outer vertex. The second is of the inner vertex.) By irreducibility of the outer walk P, there is a path from u_{src} to u_{dest} in P. Let this path be $u_{\text{src}} \longrightarrow u_1 \longrightarrow u_2 \longrightarrow \cdots \longrightarrow u_l \longrightarrow u_{\text{dest}}$ We consider two cases. Case 1: the length of the path is zero, so the path from u_{src} to u_{dest} is a self-loop. In this case $u_{\text{src}} = u_{\text{dest}}$, so v_{src} and v_{dest} are within the same outer vertex. By irreducibility of $P_{u_{\text{src}}}$, there exists a path between v_{src} and v_{dest}. The existence of a self-loop establishes that $p_{u_{\text{src}},u_{\text{src}}} \neq 0$. So the block matrix $p_{u_{\text{src}},u_{\text{src}}} P_{u_{\text{src}}}$ is not zero, establishing that there is a path in P_{cmp} between the two said vertices. Case 2: the length of the path is not zero. In this case, we can assume the existence of a path does not visit the same vertex twice and does not visit u_{src} or u_{dest} in the middle. Observe that if (u_1, v_1), (u_2, v_2) are two vertices in P_{cmp} and $u_1 \neq u_2$, then there is an edge between these two vertices iff $p_{u_1,u_2} \neq 0$. This is immediate from the fact that $p^{\text{cmp}}_{(u_1,v_1),(u_2,v_2)} = p_{u_1,u_2} \cdot s^{(u_2)}_{v_2}$ for all $u_1 \neq u_2$ and that $s^{(u_2)}$ does not have zero elements. Therefore, if $u_{\text{src}} \longrightarrow u_1 \longrightarrow u_2 \longrightarrow \cdots \longrightarrow u_l \longrightarrow u_{\text{dest}}$ is a path in P with no repeating vertices as we assumed, then $(u_{\text{src}}, v_{\text{src}}) \longrightarrow (u_1, v_1) \longrightarrow (u_2, v_2) \longrightarrow \cdots \longrightarrow (u_l, v_l) \longrightarrow (u_{\text{dest}}, v_{\text{dest}})$ is a path in P^{cmp} for *any* v_1, v_2, ...v_l within their respective outer vertices.

(3) P_{cmp} is aperiodic.

One way to prove this is by reasoning that P_{cmp} cannot have an eigenvalue of -1, which a bipartite walk matrix must have. We will instead use a combinatorial argument, in the same spirit as the proof for irreducibility. A graph is non-bipartite iff it contains a self-loop or a (not necessarily directed) cycle of odd length. By assumption, P is non-bipartite. We first assume that P has a self-loop. In this case, $p_{uu} \neq 0$ for some vertex u in P. So the uth block on the diagonal of P_{cmp}, which is $p_{uu} P_u$, is nonzero. Since P_u is also non-bipartite by assumption, P_{cmp} is non-bipartite. Next, we assume that P has a cycle of odd length. Suppose this cycle is $u_1 - u_2 - \cdots - u_l$. We've already seen in the irreducibility proof that $P^{\text{cmp}}_{(u_1,v_1),(u_2,v_2)} \neq 0$ for all u_1, u_2, v_1, v_2 such that $u_1 \neq u_2$ and $p_{u_1,u_2} \neq 0$. Then, since

all vertices in the cycle are distinct, we know that (u_1, v_1) — (u_2, v_2) — $\cdots$ — (u_l, v_l) is an odd cycle in P_{cmp}, for *any* v_1, v_2, …, v_l in their respective outer vertices.

(4) P_{cmp} is reversible.

Let $\langle s^{\text{cmp}}|$ denote the unique 1-eigenvector of P_{cmp} and let $s^{\text{cmp}}_{i,j}$ denote its elements. (Again, note that we refer to the elements of $\langle s^{\text{cmp}}|$ with two indices.) We can verify that $\langle s^{\text{cmp}}| = \begin{bmatrix} s_1\langle s^{(1)}| & s_2\langle s^{(2)}| & \cdots & s_n\langle s^{(n)}| \end{bmatrix}$. So $s^{\text{cmp}}_{i,j} = s_i s^{(i)}_j$. To show that P_{cmp} is reversible, we need to confirm that every pair of vertices (u_1, v_1), (u_2, v_2) in P_{cmp} satisfy

$$s^{\text{cmp}}_{u_1,v_1} p^{\text{cmp}}_{(u_1,v_1),(u_2,v_2)} = s^{\text{cmp}}_{u_2,v_2} p^{\text{cmp}}_{(u_2,v_2),(u_1,v_1)}. \qquad (*)$$

We divide the proof into two cases: the case where $u_1 = u_2$ and where $u_1 \neq u_2$. When $u_1 = u_2$, observe that $p^{\text{cmp}}_{(u_1,v_1),(u_2,v_2)} = p_{u_1,u_1} p^{(u_1)}_{(v_1,v_2)}$, and $p^{\text{cmp}}_{(u_2,v_2),(u_1,v_1)} = p_{u_2,u_2} p^{(u_2)}_{(v_2,v_1)} = p_{u_1,u_1} p^{(u_1)}_{(v_2,v_1)}$. If $p_{u_1,u_1} = 0$, then both sides of the above equation are equal to zero and we are done. On the other hand, when $p_{u_1,u_1} \neq 0$, equation $(*)$ is equivalent to

$$\begin{aligned} s_{u_1} s^{(u_1)}_{v_1} p_{u_1,u_1} p^{(u_1)}_{(v_1,v_2)} &= s_{u_1} s^{(u_1)}_{v_2} p_{u_1,u_1} p^{(u_1)}_{(v_2,v_1)} \\ s_{u_1} s^{(u_1)}_{v_1} p^{(u_1)}_{(v_1,v_2)} &= s_{u_1} s^{(u_1)}_{v_2} p^{(u_1)}_{(v_2,v_1)} \qquad \text{(because } p_{u_1,u_1} \neq 0\text{)} \\ s^{(u_1)}_{v_1} p^{(u_1)}_{(v_1,v_2)} &= s^{(u_1)}_{v_2} p^{(u_1)}_{(v_2,v_1)}. \qquad \text{(because } s_{u_1} \neq 0\text{)} \end{aligned}$$

Since P_{u_1} is reversible by assumption, the last equation always holds. Therefore, $(*)$ is true, which is what we wanted to prove. Next, we assume that $u_1 \neq u_2$. In this case, $p^{\text{cmp}}_{(u_1,v_1),(u_2,v_2)} = p_{u_1,u_2} s^{(u_2)}_{v_2}$, and $p^{\text{cmp}}_{(u_2,v_2),(u_1,v_1)} = p_{u_2,u_1} s^{(u_1)}_{v_1}$. So we need to check whether

$$s_{u_1} s^{(u_1)}_{v_1} \cdot p_{u_1,u_2} s^{(u_2)}_{v_2} = s_{u_2} s^{(u_2)}_{v_2} \cdot p_{u_2,u_1} s^{(u_1)}_{v_1},$$

or equivalently,

$$s_{u_1} p_{u_1,u_2} = s_{u_2} p_{u_2,u_1}.$$

Since P is reversible by assumption, we know that this equality holds. Again, this implies that equation $(*)$ is true. Therefore, P_{cmp} is reversible. □

Property 5.2.2 If

$$\lambda_1 = 1, \lambda_2, \ldots, \lambda_n$$

are eigenvalues of P, and

$$\lambda^{(j)}_1 = 1, \lambda^{(j)}_2, \ldots, \lambda^{(j)}_{m_j}$$

are eigenvalues of P_j for every $j \in \{1, \ldots, n\}$, then the eigenvalues of P_{cmp} are

$$\lambda_1, \lambda_2, \ldots, \lambda_n, \quad \text{and} \quad p_{jj}\lambda^{(j)}_2, p_{jj}\lambda^{(j)}_3, \ldots, p_{jj}\lambda^{(j)}_{m_j}, \quad \text{for every } j \in \{1, \ldots, n\}$$

(Note that $p_{jj}\lambda^{(j)}_1$ is *not* an eigenvalue.)

Proof. First, we show that $\lambda_1, \lambda_2, \ldots, \lambda_n$ are eigenvalues of P_{cmp}. If we suppose that a vector in the form of $\begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}$ is a left eigenvector of P, then it can be verified that the block vector $\begin{bmatrix} a_1\langle s^{(1)}| & a_2\langle s^{(2)}| & \cdots & a_n\langle s^{(n)}| \end{bmatrix}$ is a left eigenvector of P_{cmp} with

the same eigenvalue. So every eigenvalue of P is also an eigenvalue of P_{cmp}.

Next, we show that

$$p_{jj}\lambda_2^{(j)}, p_{jj}\lambda_3^{(j)}, \ldots, p_{jj}\lambda_{m_j}^{(j)}$$

are also eigenvalues of P_{cmp} for every $j \in \{1, \ldots, n\}$. To do so, we first prove that if $\langle v|$ is a non-1 left eigenvector of P_j, then $\langle v|u\rangle = 0$. That is, the elements of $\langle v|$ sum to zero. Since every row of P_j sums to 1, we can verify that $|u\rangle$ is a 1-*right*-eigenvector of P_j, meaning that $P_j|u\rangle = |u\rangle$. From here, we have $\langle v|u\rangle = \langle v|P_j|u\rangle = \lambda\langle v|u\rangle$, where λ is the eigenvalue of $|v\rangle$. Therefore, $(1-\lambda)\langle v|u\rangle = 0$. Since the eigenvalue of $\langle v|$ is not 1, we deduce that $\langle v|u\rangle = 0$. Having proven that $\langle v|u\rangle = 0$, we claim that the block vector $[\,0 \quad \ldots \quad 0 \quad \langle v| \quad 0 \quad \ldots \quad 0\,]$ is an eigenvector of P_{cmp}. In this vector, there are n blocks in total. The kth block is of size m_k for every $k \in \{1, \ldots, n\}$. The jth block contain $\langle v|$, a non-1 left eigenvector of P_j. All other blocks are zero. To verify that this is an eigenvector, we multiply this by P_{cmp}. Using the fact that $\langle v|u\rangle = 0$, we get

$$\begin{aligned}
&[\,0 \quad \ldots \quad 0 \quad \langle v| \quad 0 \quad \ldots \quad 0\,]P_{\text{cmp}} \\
&= [\,0 \quad \ldots \quad 0 \quad p_{jj}\langle v|P_j \quad 0 \quad \ldots \quad 0\,] \\
&= [\,0 \quad \ldots \quad 0 \quad p_{jj}\lambda\langle v| \quad 0 \quad \ldots \quad 0\,].
\end{aligned}$$

Therefore, if $\langle v|$ is an eigenvector of P_j with eigenvalue $\lambda \neq 1$, then $p_{jj}\lambda$ is an eigenvalue of P_{cmp}. This completes the proof. □

5.3 Complexity Analysis of Composed Walk

We now compare the complexities of two algorithms. The first is the "naive" method of quantum walk composition. This algorithm is an MNRS quantum walk on the outer Markov chain, where the checking procedure is MNRS quantum walks on the inner Markov chains. The second algorithm is the MNRS quantum walk on the newly proposed Markov chain characterized by P_{cmp}.

Definition 5.3.1 (Set-up, update, checking)
(1) Let S_0, the set-up cost of the outer walk, represent the cost of the unitary operation that performs the following mapping.

$$|0\rangle \mapsto \sum_{x\in\{1,\ldots,n\}} \sqrt{s_x}|x\rangle$$

(2) Let U_0, the update cost of the outer walk, represent the cost of the unitary operation that performs the following mappings and their inverses.

$$|x\rangle|0\rangle \mapsto |x\rangle \sum_{y\in\{1,\ldots,n\}} \sqrt{p_{xy}}|y\rangle$$

$$|0\rangle|y\rangle \mapsto \sum_{x\in\{1,\ldots,n\}} \sqrt{p^*_{yx}}|x\rangle|y\rangle$$

(3) Let S_1, the set-up cost of the inner walks, represent the cost of the unitary operation that performs the following mapping.

$$|z\rangle|0\rangle \mapsto |z\rangle \sum_{x\in\{1,\ldots,m_z\}} \sqrt{s^{(z)}_x}|x\rangle$$

(4) Let U_1, the update cost of the inner walks, represent the cost of the unitary operation that performs the following mappings and their inverses.

$$|z\rangle|x\rangle|0\rangle \mapsto |z\rangle|x\rangle \sum_{y\in\{1,\ldots,m_z\}} \sqrt{p^{(z)}_{xy}}|y\rangle$$

$$|z\rangle|0\rangle|y\rangle \mapsto |z\rangle \sum_{x\in\{1,\ldots,m_z\}} \sqrt{p^{(z)*}_{yx}}|x\rangle|y\rangle$$

(5) Let C_1, the checking cost, represent the cost of the unitary operation that performs the following mapping.

$$|z\rangle|w\rangle \mapsto \begin{cases} -|z\rangle|w\rangle & \text{if inner vertex } w \text{ is marked, where } w \text{ is within outer vertex } z \\ |z\rangle|w\rangle & \text{otherwise} \end{cases}$$

(6) Let ϵ_0 be the ratio of marked elements of the outer Markov chain.
(7) Let ϵ_1 be the minimum ratio of marked elements of the inner Markov chains that contain at least 1 marked element.
(8) Let δ_0 be the spectral gap of the outer Markov chain.
(9) Let δ_1 be the minimum spectral gap of the inner Markov chains that contain at least 1 marked element.

Property 5.3.2 The cost of the "naive" algorithm is in

$$O\left(\mathsf{S}_0 + \frac{1}{\sqrt{\epsilon_0}}\left(\frac{1}{\sqrt{\delta_0}}\mathsf{U}_0 + \log\frac{1}{\sqrt{\epsilon_0}}\cdot\left(\mathsf{S}_1 + \frac{1}{\sqrt{\epsilon_1}}\left(\frac{1}{\sqrt{\delta_1}}\mathsf{U}_1 + \mathsf{C}_1\right)\right)\right)\right)$$
$$= O\left(\mathsf{S}_0 + \frac{1}{\sqrt{\epsilon_0\delta_0}}\mathsf{U}_0 + \frac{1}{\sqrt{\epsilon_0}}\log\frac{1}{\sqrt{\epsilon_0}}\cdot\mathsf{S}_1 + \frac{1}{\sqrt{\epsilon_0\epsilon_1\delta_1}}\log\frac{1}{\sqrt{\epsilon_0}}\cdot\mathsf{U}_1 + \frac{1}{\sqrt{\epsilon_0\epsilon_1}}\log\frac{1}{\sqrt{\epsilon_0}}\cdot\mathsf{C}_1\right).$$

Proof. The cost of the MNRS quantum walk is in $O(\mathsf{S} + \frac{1}{\epsilon}(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C}))$. In the case of the naive algorithm, we have $\mathsf{S} = \mathsf{S}_0$, $\mathsf{U} = \mathsf{U}_0$, $\epsilon = \epsilon_0$, and $\delta = \delta_0$. As for the checking cost, the checking procedure is a quantum walk on the inner Markov chains, repeated $O(\log\frac{1}{\sqrt{\epsilon_0}})$ times for error reduction. So $\mathsf{C} \in O(\log\frac{1}{\sqrt{\epsilon_0}}(\mathsf{S}_1 + \frac{1}{\sqrt{\epsilon_1}}(\frac{1}{\sqrt{\delta_1}}\mathsf{U}_1 + \mathsf{C}_1))$. Substituting these values gives us the desired expression. □

Property 5.3.3 If $p_{jj} \leq 1 - \delta_0$ for all $j \in \{1, \ldots, n\}$, then the cost of the MNRS quantum walk on P_{cmp} is in

$$O\left(\mathsf{S}_0 + \mathsf{S}_1 + \frac{1}{\sqrt{\epsilon_0\epsilon_1}}\left(\frac{1}{\sqrt{\delta_0}}(\mathsf{U}_0 + \mathsf{U}_1 + \mathsf{S}_1) + \mathsf{C}_1\right)\right)$$
$$= O\left(\mathsf{S}_0 + \frac{1}{\sqrt{\epsilon_0\epsilon_1\delta_0}}\mathsf{U}_0 + \frac{1}{\sqrt{\epsilon_0\epsilon_1\delta_0}}\mathsf{S}_1 + \frac{1}{\sqrt{\epsilon_0\epsilon_1\delta_0}}\mathsf{U}_1 + \frac{1}{\sqrt{\epsilon_0\epsilon_1}}\mathsf{C}_1\right).$$

Proof. The cost of the MNRS quantum walk is in $O(\mathsf{S} + \frac{1}{\epsilon}(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C}))$. The parameters S, U, C, ϵ, and δ are with respect to P_{cmp}. We need to express them in terms of parameters defined in Definition 5.3.1. The set-up cost of P_{cmp} is the cost of initializing the state

$$|0\rangle \mapsto \sum_{\substack{x\in\{1,\ldots,n\}\\ y\in\{1,\ldots,m_x\}}} \sqrt{s^{\text{cmp}}_{x,y}}|x,y\rangle.$$

This can be done using the set-up procedure of the outer walk and the inner walk. As a result, we have $\mathsf{S} = \mathsf{S}_0 + \mathsf{S}_1$. The update cost of P_{cmp} is the cost of implementing the following unitary operations and their inverses.

$$|x,y\rangle|0\rangle \mapsto |x,y\rangle \sum_{\substack{z\in\{1,\ldots,n\}\\ w\in\{1,\ldots,m_z\}}} \sqrt{p^{\text{cmp}}_{(x,y),(z,w)}}|z,w\rangle$$
$$|0\rangle|z,w\rangle \mapsto \sum_{\substack{x\in\{1,\ldots,n\}\\ y\in\{1,\ldots,m_x\}}} \sqrt{p^{\text{cmp}\,*}_{(z,w),(x,y)}}|x,y\rangle|z,w\rangle$$

These unitary operations can be implemented in three steps. For example, if we want to implement the first unitary operation, we would first use the update procedure on the outer walk. By doing so, we effect the transformation $|x,y\rangle|0\rangle \mapsto |x,y\rangle\sum_{z\in\{1,\ldots,n\}}\sqrt{p_{xz}}|z,0\rangle$. Then, we check whether $x = z$. If $x = z$, we use the update operator of the inner Markov chains. Otherwise, we use the *set-up* operator of the inner Markov chains. Doing so maps the state $|x,y\rangle\sum_{z\in\{1,\ldots,n\}}\sqrt{p_{xz}}|z,0\rangle$ to

$$|x,y\rangle \left(\sqrt{p_{xx}}|x\rangle \sum_{w\in\{1,\ldots,m_x\}} \sqrt{p^{(x)}_{yw}}|w\rangle + \sum_{\substack{z\in\{1,\ldots,n\}\\ z\neq x}} \sqrt{p_{xz}}|z\rangle \sum_{w\in\{1,\ldots,m_z\}} \sqrt{s^{(z)}_{w}}|w\rangle \right),$$

which is equal to the desired state

$$|x,y\rangle \sum_{\substack{z\in\{1,\ldots,n\}\\ w\in\{1,\ldots,m_z\}}} \sqrt{p^{\mathrm{cmp}}_{(x,y),(z,w)}}|z,w\rangle,$$

as one can verify using the definition of P_{cmp}. The update cost is therefore $\mathsf{U}_0 + \mathsf{U}_1 + \mathsf{S}_1$. The checking cost of P_{cmp} is simply C_1. The ratio of marked elements of P_{cmp} is at least $\epsilon_0\epsilon_1$. This is because we assumed that every marked outer vertex contains a marked inner vertex, and the ratios of marked elements of the inner Markov chains are at least ϵ_1. Finally, the spectral gap of P_{cmp} is δ_0. By Property 5.2.2, all eigenvalues of the outer Markov chain are also eigenvalues of P_{cmp}. The magnitude of all other eigenvalues of P_{cmp} are bounded within $1-\delta_0$, because we assumed that $p_{jj} \leq 1-\delta_0$ for all $j \in \{1,\ldots,n\}$. □

Upon comparing the stated costs of the two algorithms in Property 5.3.2 and Property 5.3.3, we see that the proposed quantum walk on P_{cmp} is not necessarily more efficient. While it does not have any logarithmic factors in complexity, its U_0 and S_1 terms appear to be greater than those of the naive algorithm. We now explicitly compute the costs of these two algorithms for a specific problem, and see that the proposed algorithm is indeed less efficient in this case.

Example 5.3.4 (Triangle finding in $\tilde{O}(n^{13/10})$)

One drawback of the proposed quantum walk is that its update procedure invokes the set-up procedure of the inner Markov chains. This is disadvantageous, because set-up costs are typically much greater than update costs. This example demonstrates this drawback. Suppose we would like to determine whether a graph G contains a triangle. In [MSS07], an algorithm of triangle finding has been proposed that solves the problem with a query complexity of $\tilde{O}(n^{13/10})$, where n is the number of vertices in G. The algorithm has three layers of composition of search algorithms. The top layer is a quantum walk on a Johnson graph. Every vertex in this Markov chain is a subset of size r of vertices in G, where $r \leq n$. A vertex in this Johnson graph is considered marked if it contains two of the three vertices of a triangle. That is, if a contains a triangle *edge*. The middle layer determines whether such a triangle edge exists in the said subset, by running the Grover's algorithm over all vertices in G in search for the third vertex that completes a triangle. Finally, to determine whether a vertex completes a triangle such that one of its edges is contained in the subset of size r, we perform another quantum walk on another Johnson graph in the bottom layer. Every vertex in *this* Johnson graph is a subset of size q of the top layer Johnson graph vertex, where $q \leq r$. The bottom layer finds a triangle vertex using a method similar to the one by which the Element Distinctness algorithm finds a colliding pair. We remark that the MNRS quantum walk over the complete graph with self loops has the exact same

behaviour as Grover search. This observation allows us to express the middle layer in terms of an MNRS quantum walk for the purpose of comparison. Hence, let S_0, U_0, ϵ_0, and δ_0 represent the parameters of the top layer quantum walk, where the costs are the number of queries to the adjacency matrix of G; let S_1, U_1, ϵ_1, and δ_1 represent the parameters of the middle layer; similarly, let S_2, U_2, ϵ_2, δ_2, and C_2 represent parameters of the bottom layer. The values of these parameters are shown below.

$$\begin{array}{lll} \mathsf{S}_0 = r^2 & \mathsf{S}_1 = 0 & \mathsf{S}_2 = q \\ \mathsf{U}_0 \in O(r) & \mathsf{U}_1 = 0 & \mathsf{U}_2 \in O(1) \\ \epsilon_0 \in \Omega(r^2/n^2) & \epsilon_1 \in \Omega(1/n) & \epsilon_2 \in \Omega(q^2/r^2) \\ \delta_0 \in \Theta(1/r) & \delta_1 = 1 & \delta_2 \in \Theta(1/q) \\ & & \mathsf{C}_2 = 0 \end{array}$$

The derivation of these values, which we omit, involves a discussion about the data structures used by each layer and how they are manipulated. For our purpose of comparing methods of composition, it suffices to simply know what these values are. Generalizing Property 5.3.2 and Property 5.3.3 to the case where there are three layers of composition, the cost of the naive algorithm is in

$$\tilde{O}\left(\mathsf{S}_0 + \frac{1}{\sqrt{\epsilon_0}}\left(\frac{1}{\sqrt{\delta_0}}\mathsf{U}_0 + \left(\mathsf{S}_1 + \frac{1}{\sqrt{\epsilon_1}}\left(\frac{1}{\sqrt{\delta_1}}\mathsf{U}_1 + \left(\mathsf{S}_2 + \frac{1}{\sqrt{\epsilon_2}}\left(\frac{1}{\sqrt{\delta_2}}\mathsf{U}_2 + \mathsf{C}_2\right)\right)\right)\right)\right)\right), \tag{1}$$

whereas the cost of the quantum walk on P_{cmp} is in

$$O\left(\mathsf{S}_0 + \mathsf{S}_1 + \mathsf{S}_2 + \frac{1}{\sqrt{\epsilon_0\epsilon_1\epsilon_2}}\left(\frac{1}{\sqrt{\delta_0}}\left(\mathsf{U}_0 + \mathsf{U}_1 + \mathsf{S}_1 + \mathsf{U}_2 + \mathsf{S}_2\right) + \mathsf{C}_2\right)\right) \tag{2}$$

provided that $p_{jj} \leq 1 - \delta_0$ for all diagonal elements p_{jj} of the top layer stochastic matrix. This is true in the case of Johnson graphs. Plugging in the parameters into the above formulas, we see that Equation (1) is equal to

$$\begin{aligned} &\tilde{O}\left(r^2 + \frac{n}{r}\left(\sqrt{r}\cdot r + \left(0 + \sqrt{n}\left(0 + \left(q + \frac{r}{q}\left(\sqrt{q} + 0\right)\right)\right)\right)\right)\right) \\ &= \tilde{O}\left(r^2 + n\sqrt{r} + \frac{n^{3/2}q}{r} + \frac{n^{3/2}}{\sqrt{q}}\right), \end{aligned}$$

while Equation (2) is equal to

$$\begin{aligned} &O\left(r^2 + 0 + q + \frac{n}{r}\cdot\sqrt{n}\cdot\frac{r}{q}\left(\sqrt{r}\left(r + 0 + 0 + 1 + q\right) + 0\right)\right) \\ &= O\left(r^2 + q + \frac{n^{3/2}r^{3/2}}{q} + \frac{n^{3/2}\sqrt{r}}{q} + n^{3/2}\sqrt{r}\right). \end{aligned}$$

Equation (1) is minimized when we set $r = n^{3/5}$ and $q = n^{2/5}$, giving us the complexity $\tilde{O}(n^{13/10})$. Equation (2) is minimized when $r = n^0$ and $q = n^0$. The resulting complexity is $O(n^{3/2})$. We see that this complexity is worse than $\tilde{O}(n^{13/10})$, and it is also no better than that of the simple Grover search over all triples of vertices.

6 Conclusion

In this thesis, we investigate alternatives to the "naive" composition of the discrete time quantum walk due to Magniez, Nayak, Roland, and Santha, [MNRS11] which we refer to as the *MNRS quantum walk*. *Composition* refers to the use of search algorithms as the checking oracle of another search algorithm—the *nesting* of search algorithms. Since quantum searching generally succeeds not with certainty but with high probability, the naive method of composition introduces an extra logarithmic factor in complexity as a result from a majority voting procedure that reduces the error. Because quantum walk algorithms solve a variety of problems, we desire to eliminate this logarithmic factor. In the attempt to accomplish this goal, two approaches have been investigated.

The first method is a simplification of the existing MNRS quantum walk algorithm. This simplification, described in Algorithm 4.1.1, is to run phase estimation only once per iteration, using the same ancilla register. The benefit of this method is that if the simplification is valid, then it is possible to eliminate the logarithmic overhead in checking by using an error reduction technique described in [HMdW03]. This method of composition is the more general of the two, because it assumes no structure in the checking oracle. Since it is difficult to exactly analyze this algorithm due to the complexity of its components such as phase estimation, we tried to prove the correctness of a slightly more general algorithm, Algorithm 4.3.2, whose correctness and efficiency would imply that of Algorithm 4.1.1. However, in Section 4.5, we disproved the correctness of Algorithm 4.3.2 with an explicit counterexample. This casts doubt on whether this particular simplification of the MNRS quantum walk is valid. We leave the question of validity of this approach to future work.

The second approach is the more specific of the two. Unlike the first method, which assumes no structure in the checking oracle, we explicitly require the oracle to also be MNRS quantum walks on Markov chains. Instead of modifying the quantum walk algorithm, we tried to find a means to compose the "outer" Markov chain P, with the "inner" Markov chains on $P_1, P_2, \ldots, P_n$ into a composed Markov chain P_{cmp}, and use the existing quantum walk algorithm on this new chain. While the complexity of the resulting algorithm does not contain any logarithmic factors, it contains terms that are greater than those of the naive composition of MNRS quantum walks. Upon explicitly calculating the complexity for the Triangle Finding problem, we confirm that the complexity of the new algorithm is indeed sometimes worse than the naive method of composition. We speculate that this approach is more efficient only in cases where the set-up cost of the inner walks is small, as in the case of composed Grover search.

References

[Gro96] L. K. Grover, *A fast quantum mechanical algorithm for database search*, in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing* (ACM, 1996), pp. 212–219.

[Amb07] A. Ambainis, *Quantum walk algorithm for element distinctness*, SIAM Journal on Computing **37** (2007), no. 1, 210–239.

[MSS07] F. Magniez, M. Santha, and M. Szegedy, *Quantum algorithms for the triangle problem*, SIAM Journal on Computing **37** (2007), no. 2, 413–424.

[LG14] F. Le Gall, *Improved quantum algorithm for triangle finding via combinatorial arguments*, in *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on* (IEEE, 2014), pp. 216–225.

[NC10] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*. (Cambridge, U. K; New York: Cambridge University Press, 2010).

[Rei09] B. W. Reichardt, *Span programs and quantum query complexity: The general adversary bound is nearly tight for every boolean function*, in *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on* (IEEE, 2009), pp. 544–551.

[Bel12] A. Belovs, *Span programs for functions with constant-sized 1-certificates*, in *Proceedings of the forty-fourth annual ACM symposium on Theory of computing* (ACM, 2012), pp. 77–84.

[KLM07] P. Kaye, R. Laflamme, and M. Mosca, *An Introduction to Quantum Computing*. (Oxford: Oxford University Press, 2007).

[MNRS11] F. Magniez, A. Nayak, J. Roland, and M. Santha, *Search via quantum walk*, SIAM Journal on Computing **40** (2011), no. 1, 142–164.

[Jor75] C. Jordan, *Essai sur la géométrie à n dimensions*, Bulletin de la Société mathématique de France **3** (1875), 103–174.

[Gal08] A. Galántai, *Subspaces, angles and pairs of orthogonal projections*, Linear and Multilinear Algebra **56** (2008), no. 3, 227–260.

[GH06] A. Galantái and C. J. Hegedűs, *Jordan's principal angles in complex vector spaces*, Numerical Linear Algebra with Applications **13** (2006), no. 7, 589–598.

[Spi] M. Spivey, *A simple proof that the largest eigenvalue of a stochastic matrix is 1* (unpublished). Available at `https://mikespivey.wordpress.com/2013/01/17/eigenvalue-stochasti/`.

[HJ12] R. A. Horn and C. R. Johnson, *Matrix Analysis*. (Cambridge University Press, 2012).

[Sen06] E. Seneta, *Non-negative Matrices and Markov Chains*. (Springer Science & Business Media, 2006).

[HMdW03] P. Høyer, M. Mosca, and R. de Wolf, *Quantum search on bounded-error inputs*, in *International Colloquium on Automata, Languages, and Programming* (Springer, 2003), pp. 291–299.

[Bha13] R. Bhatia, *Matrix Analysis*, volume 169. (Springer Science & Business Media, 2013).

[KT01] A. Knutson and T. Tao, *Honeycombs and sums of hermitian matrices*, Notices of the American Mathematical Society **48** (2001), no. 2.